

Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingeniería y
Sistemas de Telecomunicación



TRABAJO FIN DE MÁSTER

***Reconocimiento de gestos basado
en acelerómetros***

Autor. Eduardo Fernández Sánchez

Ingeniero Técnico de Telecomunicación

Tutor. Wilmar Hernández Perdomo

Doctor Ingeniero

Julio 2014



RECONOCIMIENTO DE GESTOS BASADO EN
ACELERÓMETROS



Máster en Ingeniería de Sistemas y Servicios para la Sociedad de la Información

TRABAJO FIN DE MÁSTER

Título: Reconocimiento de gestos basado en acelerómetros

Autor: Eduardo Fernández Sánchez

Tutor: Wilmar Hernández Perdomo

Ponente: César Benavente Peces

Tribunal

Presidente: Amador Miguel González Crespo

Secretario: Rafael José Hernández Heredero

Vocal: José Manuel Pardo Martín

Fecha de Lectura: Julio 2014



Agradecimientos

En primer lugar, quisiera agradecer a Wilmar Hernández todo el apoyo y la confianza que siempre ha depositado en mí a lo largo de estos años juntos.

A mi familia y amigos que siempre me han animado a continuar en cualquier cosa que me he propuesto.

Y en especial a Cris, a tu lado siento que todo es posible.

Gracias a todos.



Índice

1	INTRODUCCIÓN	1
1.1.	OBJETIVOS.....	4
1.2.	ESTRUCTURA DEL TFM	4
2	RECONOCIMIENTO DE GESTOS BASADOS EN ACELERÓMETROS.....	7
2.1.	INTRODUCCIÓN	9
2.2.	RECONOCIMIENTO	9
2.2.1.	<i>Gestos</i>	10
2.2.2.	<i>Dispositivos</i>	11
2.2.3.	<i>Adquisición</i>	13
2.2.4.	<i>Segmentación</i>	14
2.2.5.	<i>Filtrado</i>	15
2.2.6.	<i>Extracción de características</i>	15
2.2.7.	<i>Clasificación</i>	16
2.3.	POSICIONAMIENTO Y NAVEGACIÓN	17
2.3.1.	<i>Hand tracking</i>	18
2.3.2.	<i>Navigation</i>	19
3	NAVEGACIÓN INERCIAL	21
3.1.	INTRODUCCIÓN (CONCEPTOS BÁSICOS)	23
3.2.	SISTEMAS DE COORDENADAS	26
3.2.1.	<i>Coordenadas body (b-frame)</i>	26
3.2.2.	<i>Coordenadas navegación (n-frame)</i>	28
3.3.	RELACIÓN ENTRE LOS SISTEMAS DE COORDENADAS	28
3.3.1.	<i>DCM</i>	28
3.3.2.	<i>Ángulos de Euler</i>	30
3.3.3.	<i>Cambio de body a navegación</i>	30

4	LÓGICA DIFUSA	35
4.1.	INTRODUCCIÓN.....	37
4.2.	CONCEPTOS BÁSICOS DE LÓGICA DIFUSA	38
4.3.	OPERACIONES	39
4.4.	SISTEMA DIFUSO	40
4.4.1.	<i>Bloque difusor</i>	40
4.4.2.	<i>Bloque de inferencia</i>	41
4.4.3.	<i>Bloque desdifusor</i>	42
5	MTI-300 AHRS	43
5.1.	ESTADOS DE FUNCIONAMIENTO	47
5.2.	COMUNICACIÓN A BAJO NIVEL.....	48
5.3.	COMUNICACIÓN A ALTO NIVEL	50
5.4.	SALIDA DE DATOS	51
6	POSICIONAMIENTO DE LA MANO.....	53
6.1.	INTRODUCCIÓN.....	55
6.2.	ADQUISICIÓN DE LOS DATOS	55
6.3.	PULSO EN LA MANO.....	56
6.4.	.FILTRADO	58
6.5.	SEGMENTACIÓN.....	60
6.6.	OBTENCIÓN DEL ÁNGULO INICIAL	63
6.7.	ACTITUD.....	64
6.7.1.	<i>Sistema de cuaterniones</i>	65
6.8.	ELIMINACIÓN DE LA GRAVEDAD	67
6.9.	CAMBIO DE COORDENADAS	69
6.10.	CORRECCIÓN DE VELOCIDAD	70
6.11.	POSICIÓN DE LA MANO	72
6.12.	CÁLCULO DE LA DISTANCIA.....	72
7	RECONOCIMIENTO POR LÓGICA DIFUSA	75
7.1.	VARIABLES DE ENTRADA	78

7.2. REGLAS DE INFERENCIA	80
7.3. SALIDA.....	82
8 PRUEBAS Y RESULTADOS	83
8.1. PRUEBAS REALIZADAS	85
8.2. RESULTADOS.....	88
9 CONCLUSIONES Y TRABAJO FUTURO	93
9.1. CONCLUSIONES	95
9.2. TRABAJO FUTURO	96
10 BIBLIOGRAFÍA.....	97
11 CÓDIGO MATLAB.....	105
11.1. MTI_ADQUISICION_DATOS.M	107
11.2. DIBUJAR_DATOS_INERCIALES.....	111
11.3. OBTENER_DATOS.M	112
11.4. FILTRO_MEDIO.M	113
11.5. SEGMENTACION_1EJE.M.....	115
11.6. SEGMENTACION_3EJES.M	116
11.7. POSICIONAMIENTO.M.....	117



Índice de figuras

FIGURA 1. GESTOS RECONOCIDOS EN [9].....	10
FIGURA 2. GESTOS RECONOCIDOS EN [19].....	10
FIGURA 3. ESQUEMA GENERAL DE RECONOCIMIENTO DE GESTOS BASADO EN ACELERÓMETROS.	11
FIGURA 4. DISPOSITIVOS CON ACELERÓMETROS.....	12
FIGURA 5. DISPOSITIVO RECONOCEDOR DE ESCRITURA ([40]).	18
FIGURA 6. PDR CON DETECCIÓN DE RAMPA ([50]).	19
FIGURA 7. UNIDADES DE MEDIDA INERCIALES.	24
FIGURA 8. SISTEMA <i>STRAPDOWN</i>	25
FIGURA 9. COORDENADAS DE LA IMU XSSENS MTi-300.	27
FIGURA 10. COORDENADAS DE <i>BODY</i> DE LA MANO.....	27
FIGURA 11. ÁNGULOS DE EULER.	30
FIGURA 12. GIRO EN YAW.	31
FIGURA 13. GIRO EN PITCH.....	31
FIGURA 14. GIRO EN ROLL.	32
FIGURA 15. LÓGICA DIFUSA Y LÓGICA CLÁSICA.	38
FIGURA 16. CONCEPTOS BÁSICOS DE LÓGICA DIFUSA.	39
FIGURA 17. ESQUEMA GENERAL DE UN SISTEMA BASADO EN LÓGICA DIFUSA.	40
FIGURA 18. FUNCIONES CARACTERÍSTICAS MÁS HABITUALES: (A) TRIANGULAR, (B) TRAPEZOIDAL, (C) GAUSSIANA Y (D) SIGMOIDAL.....	41
FIGURA 19. MTi-300 AHRS DE XSSENS.....	45
FIGURA 20. ESQUEMA GENERAL DEL MTi-300.....	45
FIGURA 21. ESTADOS DEL MTi-300.	48
FIGURA 22. CUERPO DEL MENSAJE MTComm.	49
FIGURA 23. XSSENS DEVICE API.	50
FIGURA 24. RESULTADO DE LA ADQUISICIÓN DE DATOS.....	56
FIGURA 25. A) SEÑAL DEL ACELERÓMETRO Y GIRÓSCOPO DEL EJE X SITUADO EN UNA SUPERFICIE PLANA. B) SEÑAL DE LOS MISMOS SENSORES SITUADA EN LA MANO EN REPOSO.	57
FIGURA 26. TREMOR EN LA MANO AL REALIZAR UN GESTO.	58
FIGURA 27 SEÑAL DEL ACELERÓMETRO DEL EJE X FILTRADA.	59

FIGURA 28. SEÑAL DE LOS ACELERÓMETROS CUANDO LA MANO SE MUEVE A LA DERECHA Y VUELVE AL CENTRO.	
.....	61
FIGURA 29. SEGMENTACIÓN DE LA SEÑAL DE LOS ACELERÓMETROS.....	62
FIGURA 30. ELIMINACIÓN DE LA GRAVEDAD EN LOS ACELERÓMETROS.....	68
FIGURA 31. ACELERACIÓN EN COORDENADAS DE <i>BODY</i> Y NAVEGACIÓN.	69
FIGURA 32. VELOCIDAD CORREGIDA PARA EL GESTO ADELANTE.	71
FIGURA 33. POSICIÓN DE LA MANO EN GESTO ADELANTE.....	72
FIGURA 34. BLOQUE DE LÓGICA DIFUSA REALIZADO CON FIS EDITOR.	77
FIGURA 35. VARIABLE DE SALIDA DEL SISTEMA DE LÓGICA DIFUSA.	82
FIGURA 36. POSICIÓN INICIAL.	88

Índice de tablas

TABLA 1. COMPARACIÓN DE DIFERENTES FRECUENCIAS DE MUESTREO.....	13
TABLA 2. CARACTERÍSTICAS DE LOS ACELERÓMETROS DEL MTI-300.....	46
TABLA 3. CARACTERÍSTICAS DE LOS GIRÓSCOPOS DEL MTI-300.....	47
TABLA 4. CAMPOS DEL MENSAJE MTComm.	49
TABLA 5. PARÁMETROS DE ERROR EN UNA IMU.	52
TABLA 6. VARIABLES DE ENTRADA EN EL BLOQUE DE LÓGICA DIFUSA.....	80
TABLA 7. REGLAS DE INFERENCIA.....	81
TABLA 8. DESCRIPCIÓN DE LOS GESTOS MANUALES.....	87
TABLA 9. RESULTADO DE LAS PRUEBAS (I).....	89
TABLA 10. RESULTADO DE LAS PRUEBAS (II).....	90



Resumen

En los últimos años, ha crecido de forma significativa el interés por la utilización de dispositivos capaces de reconocer gestos humanos.

En este trabajo, se pretenden reconocer gestos manuales colocando sensores en la mano de una persona. El reconocimiento de gestos manuales puede ser implementado para diversos usos y bajo diversas plataformas: juegos (Wii), control de brazos robóticos, etc.

Como primer paso, se realizará un estudio de las actuales técnicas de reconocimiento de gestos que utilizan acelerómetros como sensor de medida.

En un segundo paso, se estudiará como los acelerómetros pueden utilizarse para intentar reconocer los gestos que puedan realizar una persona (mover el brazo hacia un lado, girar la mano, dibujar un cuadrado, etc.) y los problemas que de su utilización puedan derivarse.

Se ha utilizado una IMU (Inertial Measurement Unit) como sensor de medida. Está compuesta por tres acelerómetros y tres giróscopos (MTi-300 de Xsens).

Con las medidas que proporcionan estos sensores se realiza el cálculo de la posición y orientación de la mano, representando esta última en función de los ángulos de Euler.

Un aspecto importante a destacar será el efecto de la gravedad en las medidas de las aceleraciones. A través de diversos cálculos y mediante la ayuda de los giróscopos se podrá corregir dicho efecto.

Por último, se desarrollará un sistema que identifique la posición y orientación de la mano como gestos reconocidos utilizando lógica difusa.

Tanto para la adquisición de las muestras, como para los cálculos de posicionamiento, se ha desarrollado un código con el programa Matlab. También, con este mismo software, se ha implementado un sistema de lógica difusa con la que se realizará el reconocimiento de los gestos, utilizando la herramienta FIS Editor.



Las pruebas realizadas han consistido en la ejecución de nueve gestos por diferentes personas teniendo una tasa de reconocimiento comprendida entre el 90 % y 100 % dependiendo del gesto a identificar.

Summary

In recent years, it has grown significantly interest in the use of devices capable of recognizing human gestures.

In this work, we aim to recognize hand gestures placing sensors on the hand of a person. The recognition of hand gestures can be implemented for different applications on different platforms: games (Wii), control of robotic arms ...

As a first step, a study of current gesture recognition techniques that use accelerometers and sensor measurement is performed.

In a second step, we study how accelerometers can be used to try to recognize the gestures that can make a person (moving the arm to the side, rotate the hand, draw a square, etc...) And the problems of its use can be derived.

We used an IMU (Inertial Measurement Unit) as a measuring sensor. It comprises three accelerometers and three gyroscopes (Xsens MTI-300).

The measures provided by these sensors to calculate the position and orientation of the hand are made, with the latter depending on the Euler angles.

An important aspect to note is the effect of gravity on the measurements of the accelerations. Through various calculations and with the help of the gyroscopes can correct this effect.

Finally, a system that identifies the position and orientation of the hand as recognized gestures developed using fuzzy logic.

Both the acquisition of samples to calculate position, a code was developed with Matlab program. Also, with the same software, has implemented a fuzzy logic system to be held with the recognition of gestures using the FIS Editor.

Tests have involved the execution of nine gestures by different people having a recognition rate between 90% and 100% depending on the gesture to identify.



1 Introducción

En los últimos años, ha crecido de forma significativa el interés por la utilización de dispositivos capaces de reconocer gestos humanos.

El reconocimiento de gestos manuales puede ser implementado para diversos usos y bajo diversas plataformas. En la actualidad, algunas de las implementaciones de este tipo de reconocimiento basado en la utilización de los acelerómetros como sensores de movimiento, son las siguientes:

- Juegos: Un claro ejemplo se puede encontrar en la videoconsola Wii que tiene un mando con acelerómetro. De hecho, en la mayoría de artículos consultados para la realización de este trabajo, utilizan este mando para comprobar la precisión del sistema de reconocimiento propuesto.
- Interfaz con un PC: Mediante la realización de ciertos gestos se podría sustituir el ratón de un PC por un acelerómetro colocado en la mano.
- Control de robots: Otra aplicación sería la de controlar el movimiento de un brazo robótico mediante los gestos realizados con una mano. Así, el mismo movimiento que se hiciera con la mano es el que realizaría el brazo robótico.
- Móviles: Todos los móviles de última generación incorporan un acelerómetro y, en algunos casos, giróscopos. Estos se podría utilizar para reconocer gestos hechos con el propio móvil: desde saber cuándo el usuario está hablando hasta marcar un número de teléfono dibujando el número en el aire.

Existe una gran labor investigadora en cuanto al método a utilizar para poder reconocer distintos gestos a partir de la señal proporcionada por los acelerómetros.

Uno de estos métodos se basa en la lógica difusa. Ésta se adapta mejor al mundo real en el que vivimos, e incluso puede comprender y funcionar con nuestras expresiones. Éste será el método elegido para la tarea de reconocer gestos manuales.

Pero antes del reconocimiento, se obtendrá la posición y la actitud de la mano mientras se ha realizado el gesto correspondiente. Con la ayuda de giróscopos y nociones de navegación inercial se podrá calcular a qué sitio se ha desplazado mientras ha estado en movimiento.

Por tanto, con la unión del posicionamiento de la mano y la lógica difusa se propone un reconocedor de gestos manuales.

1.1. Objetivos.

Los objetivos que se plantearon en este proyecto fueron los siguientes:

- Estudio de las diferentes técnicas existentes sobre el reconocimiento de gestos basados en acelerómetros (estado-del-arte).
- Analizar cómo afecta la gravedad en la medida de los acelerómetros.
- Corregir el efecto de la gravedad en la medida de los acelerómetros.
- Calcular la posición y la actitud de la mano.
- Con los datos de posicionamiento elaborar una técnica de reconocimiento de gestos manuales basada en lógica difusa.
- Realizar pruebas experimentales y obtener conclusiones.

1.2. Estructura del TFM

En el Capítulo 2 se explica el estado del arte actual del reconocimiento de gestos utilizando acelerómetros como sensor que mide los gestos manuales de un individuo.

En el capítulo 3 se presentan unos conceptos básicos de navegación inercial los cuales serán la base para el cálculo de la posición y orientación de la mano.

El capítulo 4 muestra unas nociones básicas sobre lógica difusa. Se realiza una pequeña introducción a cómo se componen los sistemas difusos.

Para la realización de este proyecto se ha utilizado como sensor el MTi-300 del fabricante Xsens. En el capítulo 5 se detallan todas sus características.

En el capítulo 6 se detalla todo el proceso necesario para calcular la posición y la actitud de la mano.

El reconocimiento de gestos estará basado en un sistema de lógica difusa, el cual se explica en el capítulo 7 así como una breve explicación de la misma.

El capítulo 8 presenta todas las pruebas realizadas al sistema de reconocimiento de gestos así como los resultados obtenidos de dichas pruebas.

Por último, se presentan las conclusiones de este trabajo y se plantean futuros trabajos en el capítulo 9.

2 Reconocimiento de gestos basados en acelerómetros

2.1. Introducción

Actualmente, existen muchos métodos y dispositivos para poder realizar el reconocimiento de gestos manuales. En este punto se ha querido hacer una diferenciación y se dividirán en dos partes:

- Por una parte, se encuentran los sistemas reconocedores de gestos en los que se emplean los acelerómetros como únicos sensores. Estos sistemas suelen utilizar clasificadores, modelos estadísticos, redes neuronales,... en muchos casos, se trata de sistemas que requieren aprendizaje y entrenamiento.
- Por otra, aquellos en los que los acelerómetros se utilizan junto con giróscopos y a veces incluso, con magnetómetros. Estos sistemas están más orientados al seguimiento (*hand tracking*) y en los que en ocasiones hay presentes conceptos de navegación inercial. Aun así, muchos de ellos siguen realizando la tarea del reconocimiento de gestos manuales.

2.2. Reconocimiento

Los resultados y las pruebas de los reconocedores poseen varios aspectos en común, de los cuales, existen dos muy importantes y que definen el éxito del reconocedor.

- Precisión: los métodos propuestos intentan obtener el mayor porcentaje de reconocimiento válido de un gesto. Para poder medir el grado de éxito del sistema, en la mayoría de artículos se utilizan los siguientes parámetros:

$$precision = \frac{Recognized\ Gestures}{Retrieved\ Gestures}$$

$$recall = \frac{Recognized\ Gestures}{Relevant\ Gestures}$$

- Independencia del usuario: A la hora de hacer un gesto con la mano (por ejemplo, hacer un círculo en el aire), dos personas no lo hacen

exactamente igual. Incluso, la misma persona es difícil que lo repita dos veces de la misma forma. Un sistema puede reconocer los gestos de una persona con un elevado porcentaje de aciertos pero, bajar dicho porcentaje si tiene que reconocer el mismo gesto pero hecho por otra persona diferente. Con esto, lo que se intenta cuando se diseña un sistema es que sea independiente al usuario que realice el gesto y obtenga un alto porcentaje de éxito.

En la Figura 3, se puede observar el aspecto que podría tener un sistema de reconocimiento de gestos basado en acelerómetros y cuyos bloques pasa a detallarse a continuación.

2.2.1. Gestos

La cantidad de gestos manuales que se intentan reconocer sería muy elevada para poder mostrarla en este trabajo. Algunos ejemplos de gestos serían los siguientes:

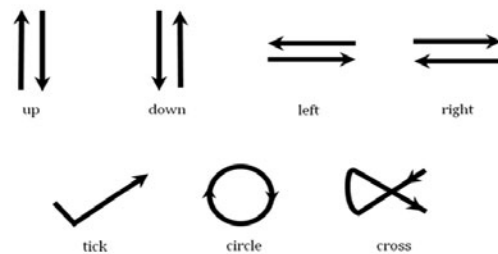


Figura 1. Gestos reconocidos en [9].

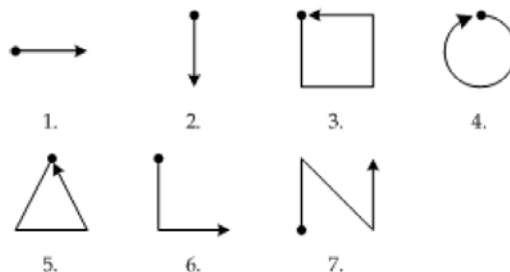


Figura 2. Gestos reconocidos en [19].

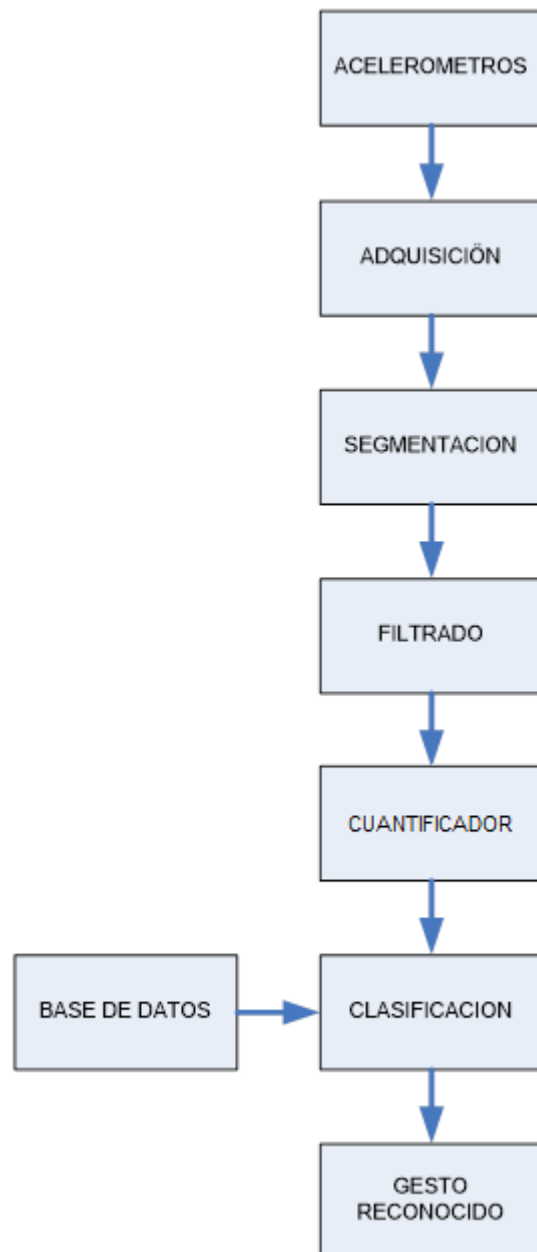


Figura 3. Esquema general de reconocimiento de gestos basado en acelerómetros.

2.2.2. Dispositivos

Para el reconocimiento de los gestos mostrados en el apartado anterior se utilizan diversos tipos de dispositivos en los que vienen integrados acelerómetros. En la siguiente figura aparecen algunos de ellos.



Figura 4. Dispositivos con acelerómetros.

En la actualidad existen diversos dispositivos comerciales en los que se integran acelerómetros, e incluso, también llevan giróscopos. Uno de los dispositivos que más aparece en las publicaciones consultadas es el mando de control de la consola Nintendo Wii, el Wiimote.

Este dispositivo está formado por un acelerómetro triaxial (ADXL330) utilizado para medir la aceleración de la mano y, junto con el sensor de infrarrojos y una barra de diodos LEDs infrarrojos exterior, poder determinar con exactitud los movimientos de la mano así como su intensidad mientras se juega.

Los tipos de dispositivos que cuentan con acelerómetros más comunes son los siguientes:

- Acelerómetro triaxial LIS30V32DQ de STMicroelectronics: [1], [31]
- Mando de la Wii: [6], [8], [12], [13],[18], [21], [22], [24], [25]
- Acelerómetro triaxial de Analog Devices ADXL330: [17]
- IMU de Xsens MTx: [18]
- Teléfono móvil: [18], [23], [30]
- ADIS 16350 IMU: [26]
- Guante con acelerómetros: [39], [53], [54]

2.2.3. Adquisición

La señal de salida de los acelerómetros es una señal continua cuya tensión es proporcional a la aceleración que se produce en los mismos. Para el manejo de la información que nos proporcionan estos sensores esta señal se debe pasar por un conversor ADC para obtener una señal discreta.

En general, en la mayoría de la documentación consultada la frecuencia de muestreo no es suele ser elevada, pudiéndose establecerse un valor medio en la frecuencia de muestreo de 100 H z. De hecho, muchos sistemas de reconocimiento optan por fijar dicho valor entre 20 y 50 Hz.

La razón de utilizar estos valores de frecuencia de muestreo es para intentar reducir lo máximo posible el tiempo de compilación del procesador encargado de realizar los cálculos pertinentes para el reconocimiento.

Frecuencias (Hz)	200 Hz	100 Hz	50 Hz
Tasa de acierto (%)	88.46	84.32	79.14
Tiempo compilación (s/ 7 gestos)	0.796	0.711	0.670

Tabla 1. Comparación de diferentes frecuencias de muestreo.

Pero por otro lado, la Tabla 1 muestras las pruebas que se realizaron en **¡Error! No se encuentra el origen de la referencia.**, variando la frecuencia de muestreo. Lo que refleja es que la tasa de acierto disminuye cuando baja la frecuencia de muestreo. Esto puede ser debido a que si los gestos se han realizado a una velocidad elevada y, la frecuencia de muestreo no es lo suficientemente grande, es posible perder parte de la información del gesto.

Por tanto, una baja tasa de reconocimiento puede estar ligada a la frecuencia de muestreo.

2.2.4. Segmentación

En este paso lo que se pretende es aislar de la señal obtenida de los acelerómetros la información perteneciente a un posible gesto realizado discriminando el resto. Aquí no se comprueba qué tipo de gesto es o qué características presenta la señal ya que eso será tarea de los procesos posteriores. En los procesos de reconocimiento se utilizan varias técnicas de segmentación que se podrían dividir en dos tipos:

- **Manual:** El usuario pulsa un botón, una tecla,... para indicar que una vez la pulse comenzará a realizar un gesto. Cuando suelte el botón el gesto habrá concluido. En este caso toda la información que se obtiene es del gesto realizado, no teniendo que discriminar ninguna información adicional. Este método es muy utilizado en los sistemas que utilizan el mando Wiimote ya que aprovechan los botones que incluye el dispositivo
- **Automática:** El sistema estaría “vigilando” la señal de entrada (acelerómetros) y basándose en algún tipo de reglas (estableciendo umbrales de la amplitud, comparando el dato actual con el anterior,...), detectaría el principio de un posible gesto. Toda la información a partir de este punto será procesada por los distintos bloques del sistema hasta que se detecte el final del gesto.

Los diferentes modos de segmentación serán los siguientes:

- Pulsar un botón: [1], [8], [12], [13], [18], [24], [25]
- Umbrales: [6], [27]
- Comprobación de parámetros de la señal: [9]
- “Auto-cut”(umbrales):[10]
- Sliding Windows: [23]

2.2.5. Filtrado

Los acelerómetros suelen tener a su salida cierta componente de ruido de alta frecuencia que se eliminará con la utilización de un filtro. De esta forma se tendrá una señal más limpia por lo que será más fácil reconocer posibles gestos dentro de ella.

Las diferentes opciones de filtrado pueden ser:

- Filtrado de la media de k valores: [1]
- Filtrado alta frecuencia: [9], [22]
- DWT: [10]
- Zero-bias compensation: [10]
- Temporal compression: [12]
- Filtro de suavizado de k puntos: [19]
- Sliding Window: [23], [26]

2.2.6. Extracción de características

En algunos sistemas no se trabajará directamente con la señal obtenida de los acelerómetros, sino que de ésta se extrae cierta información: su media, su varianza, la correlación con la información de otro eje, se le aplican transformadas (DCT, FFT),... De esta forma, se reconocerá el gesto no por los valores temporales de aceleración, sino por la información que de estas transformaciones se desprende.

Algunos métodos empleados son:

- FFT (Fast Fourier Transform): [21]
- DCT (Discrete Cosine Transform): [10]

A la hora de realizar el reconocimiento de gestos la mayoría de los algoritmos utilizados realizan un número elevado de operaciones por lo que el gasto computacional puede llegar a ser elevado. La función de este proceso es la de reducir

el número de muestras a procesar. Unos de los algoritmos más utilizados con este fin es el K-means ya que realiza la tarea de agrupamiento de datos (clustering). En otros casos, se puede reducir la cantidad de datos codificando los mismos según diversas reglas (pueden ser codificaciones lineales o no lineales)

Algunos ejemplos pueden ser:

- Algoritmo K-means: [1], [25], [30]
- Algoritmo k-NN: [6], [19]
- Codificación por signos: [9]
- Affinity Propagation (cluster): [12]
- Codificación en niveles: [13]

2.2.7. Clasificación

En este proceso, con la información obtenida de los procesos anteriores, es donde se reconoce el gesto.

Aquí, hay una gran variedad de algoritmos y procedimientos que realizan esta tarea: HMM (Hidden Markov Models), DTW (Dynamic Time Warping), ANN (Artificial Neural Networks), PCA (Principle Component Analysis), SVM (Support Vector Machine), algoritmo AdaBoost,...

Muchos de estos métodos utilizan una base de datos que es necesaria para entrenar al sistema, debe aprender a reconocer. Esta base de datos debe ser lo más heterogénea posible si el objetivo es reconocer los gestos de distintas personas.

Algunos de los más utilizados son los siguientes:

- Modelos ocultos de Markov (HMM): [1], [10], [18], [21], [25], [28], [30], [47].
- HMM-based threshold model: [20]
- DTW (Dynamic time warping): [6], [12], [13], [23], [26], [27], [28]

- Red Neuronal Recurrente (red Hopfield): [9]
- ANN (Artificial Neural Network): [21]
- ANN (con algoritmo de aprendizaje back-propagation): [17]
- SVM (Support Vector Machine): [18], [27]
- PCA (Principal Component Análisis): [19], [22]
- FDSVM (Frame-based Descriptor and multi-class SVM): [24]
- Lógica difusa: [31], [46], [53]

En la tarea del reconocimiento también son utilizados ciertos clasificadores tales como:

- Naive bayes: [1], [23], [25], [30]
- Algoritmo propio: [5]
- Clasificador lineal simple: [8]
- Algoritmo AdaBoost: [8], [20]
- Sign sequence and template matching: [9]
- ATM (adaptive threshold model): [20]

2.3. Posicionamiento y navegación

Los acelerómetros también son utilizados como parte de las investigaciones relacionadas con el posicionamiento y la navegación.

En este tipo de sistemas también aparecen otro tipo de sensores como son los giróscopos y los magnetómetros, que se complementan con los acelerómetros para, mediante ciertos cálculos, ofrecer los datos de orientación de un objeto, una mano, un individuo...

Dichos cálculos difieren a lo visto en los apartados anteriores por lo que se ha optado por separarlos. Para llegar a obtener valores de orientación y posicionamiento los métodos utilizados a veces se corresponden con conceptos propios de la navegación inercial más que con el concepto de modelos estadísticos, algoritmos de aprendizaje...

Por tanto, aunque en algunos detalles puedan ser similares, el esquema general presentado con anterioridad (Figura 3) ya no valdría para esta sección.

Aun así, el objetivo del reconocimiento de gestos manuales sigue presente y se mostrarán algunos ejemplos de aplicaciones que lo realizan pero de una forma distinta a la de los apartados anteriores.

2.3.1. Hand tracking

Otra forma de reconocer los gestos manuales a través del seguimiento de la mano. Así, en [39] el objetivo no sólo es reconocer ciertos gestos sino que además, hacerlo calculando su orientación a través de una matriz de rotación.

En el caso de [40] se utiliza un dispositivo con aspecto de bolígrafo que llevará insertados sensores inerciales (acelerómetros y giróscopos). Se pretende reconocer los gestos hechos al escribir con el bolígrafo (números del 0 al 9), es decir, reconocer la escritura de ciertos signos. Para ello lo que se intenta es calcular la posición y actitud del bolígrafo para poder interpretar los cambios en éstas como escritura de signos a través del método EPD (*End-Point-Detection*).



Figura 5. Dispositivo reconocedor de escritura ([40]).

Además de reconocer gestos, existen trabajos en los que se intenta conocer de la forma más precisa la actitud de la mano la posición de la mano [41], [42], [52], como su actitud [5], [43], [45], [49].

2.3.2. Navigation

Ya se ha introducido el concepto de navegación inercial (se verá en detalle en el capítulo 3) que será de especial interés en este trabajo.

En este campo, lo que se intenta es poder conocer la actitud y posición de un móvil a lo largo del tiempo. Se basa en la utilización de sensores inerciales, acelerómetros y giróscopos.

Evitar errores en el cálculo de la posición no es una tarea fácil pues algunos de estos errores con el tiempo crecen de manera considerable. Un método para poder evitarlos es utilizar el Filtro de Kalman ([50]) . También es ampliamente utilizado el Filtro de Kalman Extendido (EKF) ([48]).

Un ejemplo de aplicación lo tenemos en [50] en el que lo que se pretende es utilizar un sistema de navegación inercial para realizar el seguimiento de un peatón habiéndole colocado los sensores en su pie, tal y como se ve en la siguiente figura.



Figura 6. PDR con detección de rampa ([50]).

3 Navegación inercial

3.1. Introducción (conceptos básicos)

Desde los inicios de la navegación se han propuesto distintas técnicas para lograr obtener con la mayor precisión posible la posición, velocidad y orientación de un vehículo.

Se presentan ciertos conceptos básicos en la navegación inercial definidos por *Mohinder S.Grewal et al.* ([37]):

- Sistema de Referencia Inercial: es un sistema de coordenadas en el que son válidas las leyes de movimiento de Newton. Los sistemas de referencia inercial ni rotan ni aceleran.
- Sensores Inerciales IMU (Inertial Measuring Unit): miden la variación de rotación (giróscopos) y la aceleración (acelerómetros), que quedan definidas como vectores.
- Navegación Inercial: utiliza giróscopos y acelerómetros para mantener una estimación de la posición, velocidad, actitud y variación de actitud del vehículo en el que está montado el sistema de navegación inercial.
- Sistema de Navegación Inercial (Inertial Navigation System, INS): estará formado por:
 - Una Unidad de Medida Inercial (Inertial Measurement Unit, IMU) o una Unidad de Referencia Inercial (Inertial Reference Unit) que incluye un conjunto de sensores: acelerómetros y giróscopos ligados a una plataforma común para mantener las mismas orientaciones relativas.
 - Uno o más procesadores de navegación para procesar estas las medidas de los sensores y calcular la posición y la orientación.

Existen diferentes diseños de sistemas de navegación inercial con diferentes características de funcionamiento pero, en general, se pueden clasificar en dos categorías (Figura 7):

- Gimbaled: es un sistema de referencia rígido cuya misión es la de aislar el conjunto de los sensores de los movimientos de rotación externos. En la práctica es muy difícil aislar completamente al sistema de las perturbaciones exteriores.
- Strapdown: en este tipo de sistemas los sensores se encuentran alineados con los ejes del cuerpo donde están instalados. Son sistemas que requieren de una mayor capacidad de cálculo ya que es necesario calcular las ecuaciones del movimiento para 6 grados de libertad (3 acelerómetros y tres giróscopos). Será este tipo de sistema el que se utilizará en este trabajo mediante la utilización de la IMU de Xsens MTi-300.

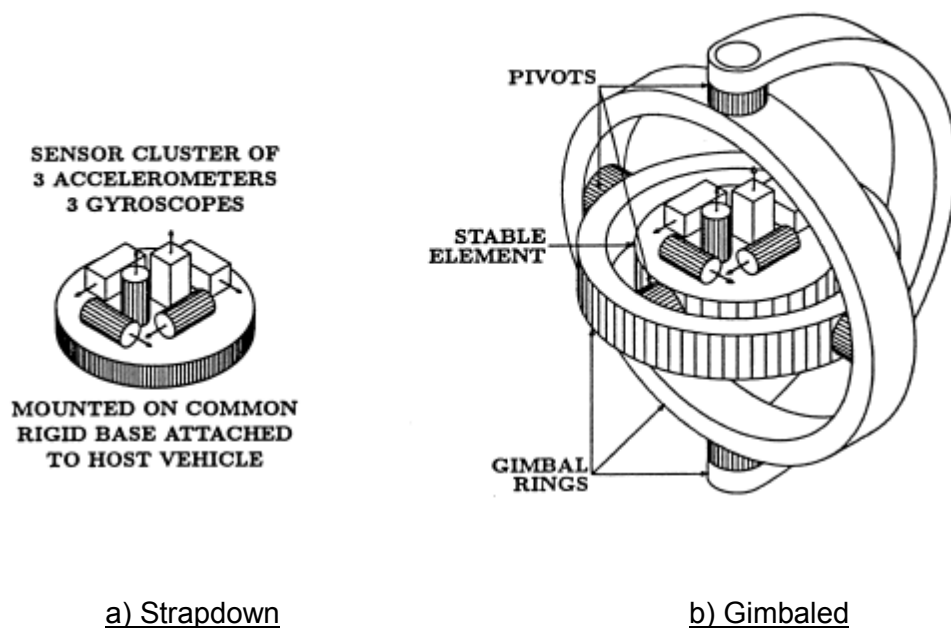


Figura 7. Unidades de medida inerciales.

Sistema de navegación *strapdown* que consiste en que los sensores están montados sobre el dispositivo.

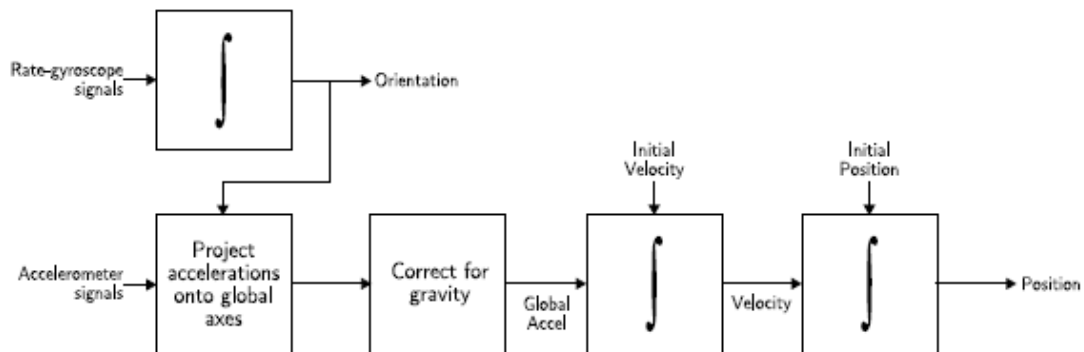


Figura 8. Sistema *strapdown*.

Los giroscopios entregan a su salida la velocidad angular ($^{\circ}/s$) a la que gira el objeto al que están sujetos. Mediante la integración de la velocidad angular se obtiene la orientación en grados del objeto.

A través de la orientación se puede conocer cuál será el efecto de la gravedad en el objeto.

Midiendo la gravedad en los acelerómetros cuando estos están alineados con los ejes globales, en el eje perpendicular al horizonte se tendría un valor de $1g$ y de $0g$ en los otros dos ejes.

Cuando el objeto ha variado su posición (en uno o más ejes) debido a un movimiento, a priori no se puede saber cuál será el valor de la aceleración en cada uno de sus ejes. Esto es debido a que el valor de la aceleración estará compuesto por la aceleración correspondiente al movimiento y la aceleración debida a la gravedad.

Conocida la orientación en grados que ha proporcionado la integración de los giróscopos sí se puede calcular el efecto de la gravedad, ya que sabiendo los ángulos de inclinación, se puede calcular el valor proporcional de la gravedad que afecta al acelerómetro.

De esta forma, se puede eliminar el efecto que tiene la gravedad sobre los acelerómetros y es la que se utilizará más adelante.

3.2. Sistemas de coordenadas

Varios sistemas de coordenadas de referencia están disponibles para un diseñador de un sistema de navegación inercial para su uso en los cálculos de navegación de posición, velocidad y actitud.

La elección del sistema de coordenadas apropiado dependerá, en gran medida, de los requisitos de la misión, la facilidad de implementación, el almacenamiento de datos y su velocidad, y de la complejidad de las ecuaciones de navegación.

En general, existen seis sistemas de coordenadas fundamentales de interés para expresar el movimiento en relación con algún marco de referencia ([32]):

- *True Inertial frame* (I).
- *Earth-Centered Inertial (ECI) Frame* (i).
- *Earth-Centered Earth-Fixed (ECEF) Frame* (e).
- *Navigation Frame* (n).
- *Body Frame* (b).
- *Wander-Azimuth Frame* (c)

En este proyecto se utilizarán las coordenadas de *body* y de navegación que serán descritas en los siguientes apartados.

3.2.1. Coordenadas *body* (*b-frame*)

Este sistema de coordenadas es normalmente utilizado en los sistemas de navegación inercial *strapdown* en los que los ejes coinciden con el centro de masas de un vehículo.

Las coordenadas de la IMU son las que se especifican en la siguiente figura:

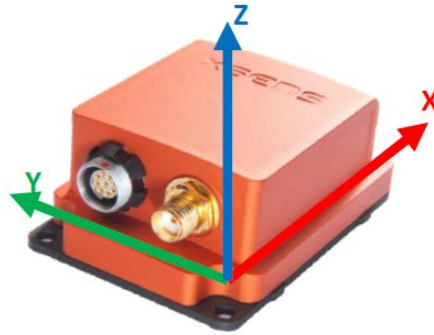


Figura 9. Coordenadas de la IMU XSens MTi-300.

La situación de los ejes en la IMU es la siguiente:

- Eje x: apunta hacia adelante.
- Eje y: apunta hacia la izquierda.
- Eje z: apunta hacia arriba.

Sin embargo, las coordenadas de *body* de la mano serán las que se muestran en la Figura 10 y cuyos ejes quedan definidos de la siguiente forma:

- Eje x: apunta hacia adelante.
- Eje y: apunta hacia la derecha.
- Eje z: apunta hacia abajo.

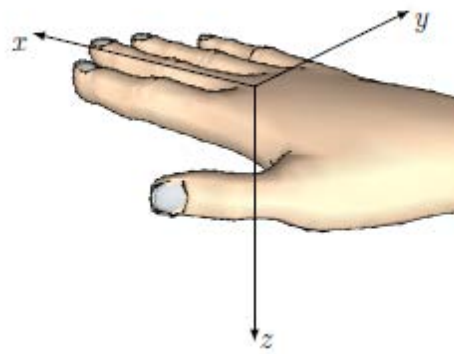


Figura 10. Coordenadas de *body* de la mano.

Para pasar de los datos proporcionados por los sensores de la IMU (tanto acelerómetros como giróscopos) a coordenadas de *body* habrá que realizar los siguientes cambios:

$$a^b = C_{acc}^b \cdot a^{acc}$$

$$gyr^b = C_{gyr}^b \cdot gyr^{gyr}$$

$$C_{gyr}^b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

$$C_{acc}^b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

3.2.2. Coordenadas navegación (n-frame)

Navigation frame tiene su origen en la ubicación del sistema de navegación inercial.

También es conocido como NED (North, East, Down), ya que los ejes X e Y son tangentes a la superficie de la tierra. El eje X apunta hacia el norte, el eje Y hacia el este y el eje Z hacia el centro de la tierra.

3.3. Relación entre los sistemas de coordenadas

La orientación en los dos sistemas de coordenadas vistos ahora se va representar mediante los ángulos de Euler y la relación entre ellos quedará definida por la Matriz de Direcciones Coseno.

3.3.1. DCM

La Matriz de Direcciones Coseno (Direction Cosine Matrix, DCM) está definida de la siguiente forma:

$$C_b^a = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}$$

La Matriz de Direcciones Coseno se puede identificar como un sistema de cambio de coordenadas en el que teniendo un marco de referencia a se pasa al marco b a través de la rotación de un vector.

Dado que los componentes de cualquier vector unitario son simplemente sus proyecciones sobre las direcciones unitarias del sistema de referencia empleado, cada uno de estos puede escribirse como el producto punto de un par de vectores unitarios [56]. Es por ello que los componentes de una matriz de rotación son referidos frecuentemente como cosenos de dirección - por definición el producto punto de dos vectores unitarios resulta en el coseno del ángulo entre estos:

$$v_1 \cdot v_2 = \cos(a)$$

Donde v_1 y v_2 son vectores unitarios en $\mathbb{R}^{3 \times 1}$ con un mismo origen y a el ángulo entre ellos.

Por tanto, para realizar un cambio de coordenadas se emplearía la siguiente expresión

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}^a = C_b^a \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}^b$$

Las propiedades más importantes de la matriz de cambio de coordenadas son las siguientes:

$$C_n^b = C_\psi C_\theta C_\phi$$

$$(C_b^a)^{-1} = (C_b^a)^T = C_b^a \quad (1)$$

3.3.2. Ángulos de Euler

Los ángulos de Euler constituyen un conjunto de tres coordenadas angulares, que sirven para especificar la orientación de un sistema de referencia de ejes ortogonales, normalmente móvil, respecto a otro sistema de referencia de ejes ortogonales normalmente fijos.

Estos ángulos quedan definidos de la siguiente manera (Figura 11):

- Φ = 'roll' = rotación alrededor del eje X, definido por $[-180^\circ$ a $180^\circ]$.
- θ = 'pitch' = rotación alrededor del eje Y, definido por $[-90^\circ$ a $90^\circ]$.
- ψ = 'yaw' = rotación alrededor del eje Z, definido por $[-180^\circ$ a $180^\circ]$.

Con estos ángulos y, mediante una sucesión ordenada de giros, se puede definir un cambio de sistemas de coordenadas.

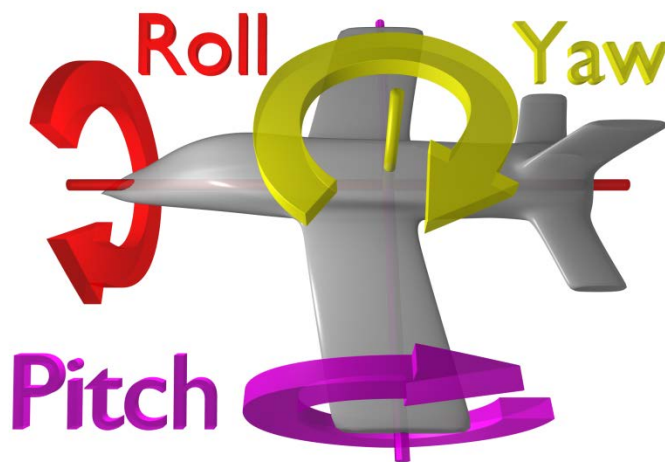


Figura 11. Ángulos de Euler.

3.3.3. Cambio de *body* a navegación

Para definir el cambio de coordenadas de *body* a *navigation* se deben realizar una sucesión de giros ordenados tal y como se muestra a continuación.

Giro en Yaw

En primer lugar, se realiza un giro sobre el ángulo de yaw, de esta forma, se pasa del sistema de coordenadas (x, y, z) al nuevo sistema (x', y', z') .

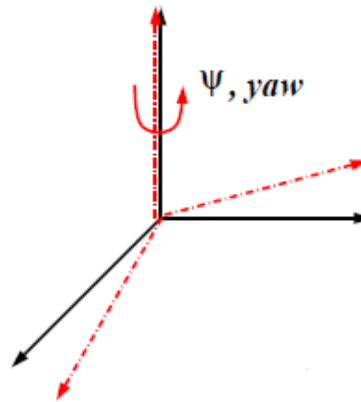


Figura 12. Giro en Yaw.

$$C_{\psi} = \begin{bmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

Giro en Pitch

En segundo lugar se realiza un giro sobre el ángulo pitch con el que se consigue cambiar del sistema (x', y', z') al sistema (x'', y'', z'') .

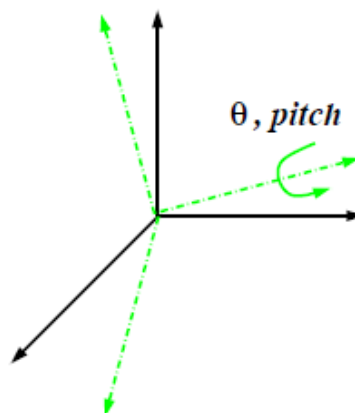


Figura 13. Giro en Pitch.

$$C_{\theta} = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix} \quad (3)$$

Giro en Roll

Y para finalizar, se realiza el tercer giro sobre el ángulo de roll para conseguir pasar de (x'', y'', z'') a (X, Y, Z) .

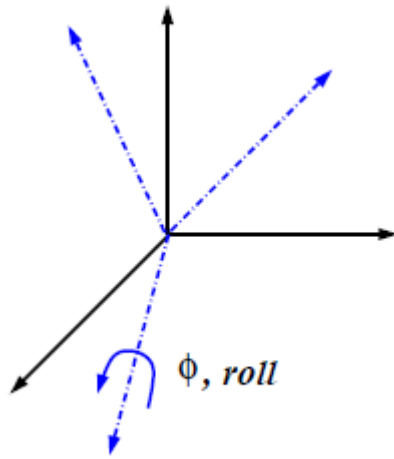


Figura 14. Giro en Roll.

$$C_{\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix} \quad (4)$$

A raíz de estos giros y, con la ecuaciones (2), (3) y (4), se puede definir la matriz de cambio de coordenadas con la siguiente expresión:

$$C_n^b = C_{\psi} C_{\theta} C_{\phi} \quad (5)$$

$$C_n^b = \begin{bmatrix} \cos \theta \cos \psi & \cos \theta \sin \psi & -\sin \theta \\ \cos \psi \sin \theta \cos \phi - \sin \psi \cos \phi & \cos \psi \cos \phi + \sin \psi \sin \theta \sin \phi & \cos \theta \sin \phi \\ \cos \psi \sin \theta \cos \phi + \sin \psi \sin \phi & \sin \psi \sin \theta \cos \phi - \cos \psi \sin \phi & \cos \theta \cos \phi \end{bmatrix}$$

La ecuación (5) corresponde a la matriz de cambio de coordenadas de navegación a *body*. Pero en realidad, lo que se pretende es pasar de coordenadas de *body* a *navigation*, por lo que si aplicamos (1) se obtiene que:

$$C_b^n = (C_n^b)^T \quad (6)$$

4 Lógica difusa

En este proyecto, se ha desarrollado un sistema basado en la lógica difusa para poder realizar el reconocimiento de los gestos manuales realizados por una persona.

En el apartado 7 se darán detalles de este sistema pero antes se van a explicar unos conceptos básicos de lógica difusa con el fin de comprender el sistema realizado.

4.1. Introducción

El término lógica difusa (*fuzzy logic*) fue por primera vez utilizado por Lofti A. Zadeh [57] en 1965.

Para intentar explicar el concepto de conjunto difuso, Zadeh utilizó el ejemplo de los hombres altos. Así, si se define el conjunto de los hombres altos según la teoría clásica, pertenecerían a este conjunto los hombres cuya altura estuviera a partir de un determinado valor que, por ejemplo se puede situar en 1,80 metros. Todos los demás hombres quedarían fuera de este conjunto. De esta forma, si un hombre mide 1,81 m formaría parte del conjunto de los hombres altos y, un hombre de 1,79, no formaría parte de este conjunto.

Sin embargo, no parece muy lógico decir que un hombre es alto y otro no lo es cuando su altura difiere en dos centímetros. El enfoque de la lógica difusa considera que el conjunto “hombres altos” es un conjunto que no tiene una frontera clara para pertenecer o no pertenecer a él: mediante una función que define la transición de “alto” a “no alto” se asigna a cada valor de altura un grado de pertenencia al conjunto, entre 0 y 1. Así por ejemplo, un hombre que mida 1.79 podría pertenecer al conjunto difuso “hombres altos” con un grado 0.8 de pertenencia, uno que mida 1.81 con un grado 0.85, y uno que mida 1.50 m con un grado 0.1.

Visto desde esta perspectiva se puede considerar que la lógica clásica es un caso límite de la lógica difusa en el que se asigna un grado de pertenencia 1 a los hombres con una altura mayor o igual a 1.80 y un grado de pertenencia 0 a los que tienen una altura menor.

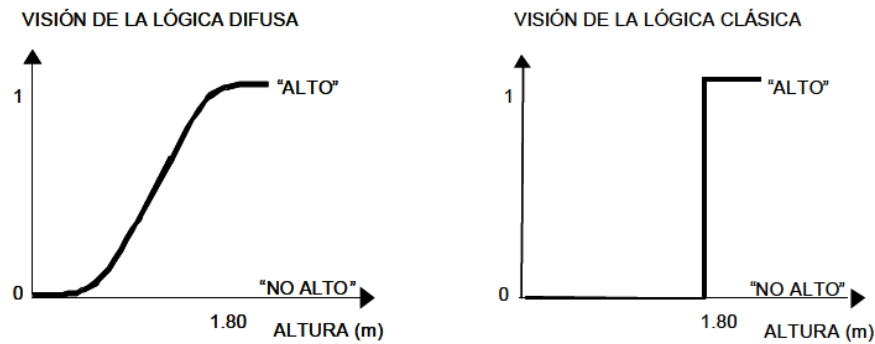


Figura 15. Lógica difusa y lógica clásica.

Cada elemento presenta un grado de pertenencia a un conjunto difuso que puede tomar cualquier valor entre 0 y 1. Este grado de pertenencia se define mediante la función característica asociada al conjunto difuso: para cada valor que pueda tomar un elemento o variable de entrada x la función característica $\mu_A(x)$ proporciona el grado de pertenencia de este valor de x al conjunto difuso A .

4.2. Conceptos básicos de lógica difusa

Se van a definir unos conceptos básicos relativos a la lógica difusa [55]:

- Universo de discurso (U): es el intervalo de todos los posibles valores aplicables a una variable de un sistema.
- Entrada real: son las entradas provenientes del mundo exterior hacia el sistema difuso.
- Etiqueta: es el nombre descriptivo usado para identificar una función de membresía.
- Dominio: es el intervalo de valores sobre el cual se ubica el ancho de una función de membresía.
- Grado de membresía o pertenencia: es el grado de compatibilidad de una entrada con una función de membresía en un intervalo de 0 a 1. El cero es para no pertenencia, el 1 es para pertenencia total y los valores intermedios para pertenencia parcial.

- Funciones de membresía o pertenencia (μ): función matemática que nos indica el grado de pertenencia de las variables de entrada y de salida.

En la siguiente figura se puede apreciar estos conceptos básicos de una forma gráfica.

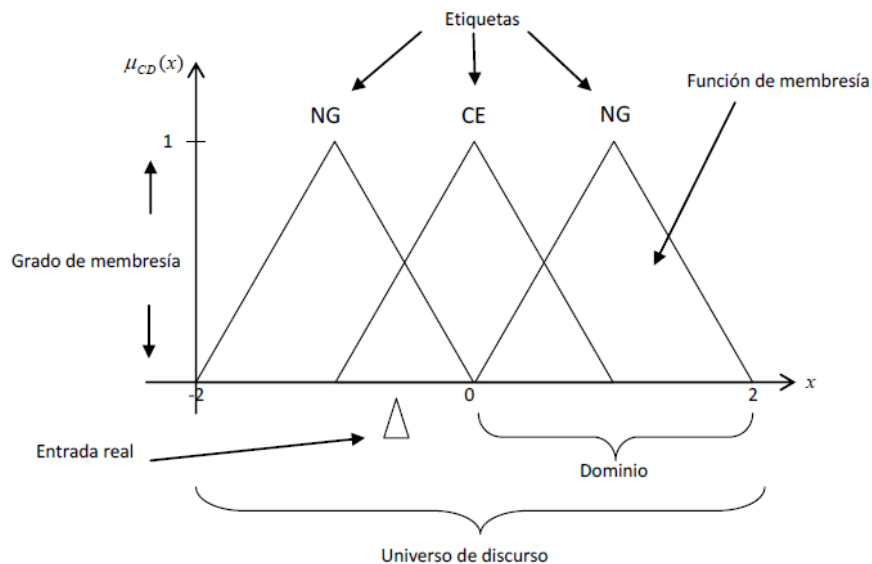


Figura 16. Conceptos básicos de lógica difusa.

4.3. Operaciones

Las operaciones básicas entre conjuntos difusos son las siguientes:

- Complementario: Sea un conjunto difuso A se podrá decir que \bar{A} es su complementario si su función característica está definida de la siguiente forma:

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x)$$

- Unión: Teniendo los conjuntos difusos A y B , se define su unión $A \cup B$ como un conjunto difuso en el universo de discurso U , siendo la función de pertenencia la mostrada a continuación:

$$\mu_{A \cup B}(x) = \max[\mu_A(x), \mu_B(x)]$$

- Intersección: Teniendo los conjuntos difusos A y B, se define su intersección $A \cap B$ como un conjunto difuso en el universo de discurso U, siendo la función de pertenencia la mostrada a continuación:

$$\mu_{A \cap B}(x) = \min[\mu_A(x), \mu_B(x)]$$

Estas tres operaciones definidas para conjuntos difusos cumplen, al igual que en la teoría clásica de conjuntos, asociatividad, conmutatividad y distributividad así como las leyes de Morgan.

4.4. Sistema difuso

Después de tener una idea de los conceptos básicos de lógica difusa en la siguiente figura se muestra un esquema general de un sistema basado en lógica difusa.

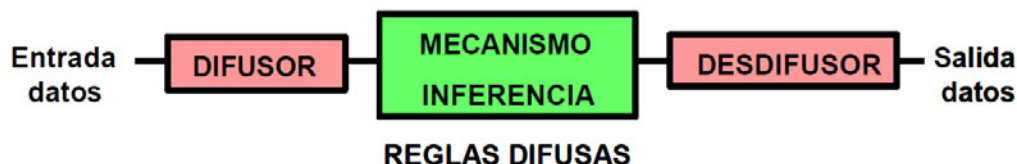


Figura 17. Esquema general de un sistema basado en lógica difusa.

En los siguientes subapartados se explicará las funciones de cada uno de los bloques representados en la figura anterior.

4.4.1. Bloque difusor

En este bloque a cada variable de entrada se le asigna un grado de pertenencia a cada uno de los conjuntos difusos que se ha considerado, mediante las funciones características asociadas a estos conjuntos difusos. Las entradas a este

bloque son valores concretos de las variables de entrada y las salidas son grados de pertenencia a los conjuntos difusos considerados.

En la siguiente figura se presentan algunas de las funciones características más utilizadas.

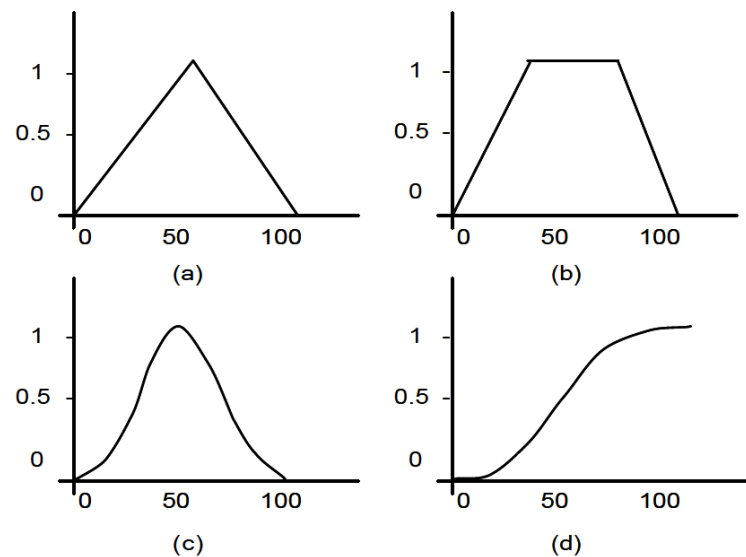


Figura 18. Funciones características más habituales: (a) triangular, (b) trapezoidal, (c) gaussiana y (d) sigmoideal.

En este bloque, es donde se asignan etiquetas lingüísticas a las funciones características. Si se continuara con el ejemplo del hombre alto, pero esta vez que mida la altura de todos los hombres, se podrían poner etiquetas como “alto”, “bajo”, “muy bajo”...

A la hora de definir las funciones características con sus correspondientes etiquetas, la experiencia del experto en el proceso será clave para un correcto funcionamiento del sistema.

4.4.2. Bloque de inferencia

Bloque que, mediante los mecanismos de inferencia, relaciona conjuntos difusos de entrada y de salida y que representa a las reglas que definen el sistema.

Las reglas se definen con la forma siguiente:

SI <proposición difusa a> **ENTONCES** <proposición difusa b>

Las entradas a este bloque son conjuntos difusos (grados de pertenencia) y las salidas son también conjuntos difusos, asociados a la variable de salida.

A la *proposición difusa a* de la izquierda se denomina *antecedente* o *premisa* y la *proposición difusa b* se conoce como *consecuente* o *conclusión*. Una regla difusa representa una relación difusa entre el antecedente y el consecuente, cuya función de pertenencia viene dada por la expresión:

$$\mu_{A \rightarrow B}(x, y) = \Phi[\mu_A(x), \mu_B(x)]$$

Donde Φ se denomina operador de implicación y puede ser representado por distintas funciones.

4.4.3. Bloque desdifusor

A partir del conjunto difuso obtenido en el mecanismo de inferencia y mediante los métodos matemáticos de desdifusión (*defuzzification*), se obtiene un valor concreto de la variable de salida, es decir, el resultado.

Los métodos más empleados son los siguientes:

- Centro de área, también conocido como Centro de Gravedad o Centroide.
- Centro del Área Mayor.
- Máximo.
- Centro de Sumas.
- Media Difusa, también conocido como método de la *altura*.

5 MTi-300 AHRS

Para la realización de este trabajo se ha utilizado una IMU (Inertial Measurement Unit) del fabricante Xsens cuyo modelo es el MTi-300 AHRS.



Figura 19. MTi-300 AHRS de Xsens.

Dicho sensor de medición inercial cuenta con acelerómetros 3D, giroscopios 3D y con magnetómetros 3D. Además, posee un DSP (Digital Signal Processor) para realizar determinados cálculos con los datos de los sensores: es capaz de calcular en tiempo real la actitud del sensor, así como las salidas calibradas en los tres ejes, de la aceleración lineal y velocidad angular. El esquema general puede verse en la Figura 20.

Proporciona datos de roll, pitch y yaw orientado al norte magnético, sin deriva. Por esto, se le denomina AHRS, Attitude and Heading Reference System.

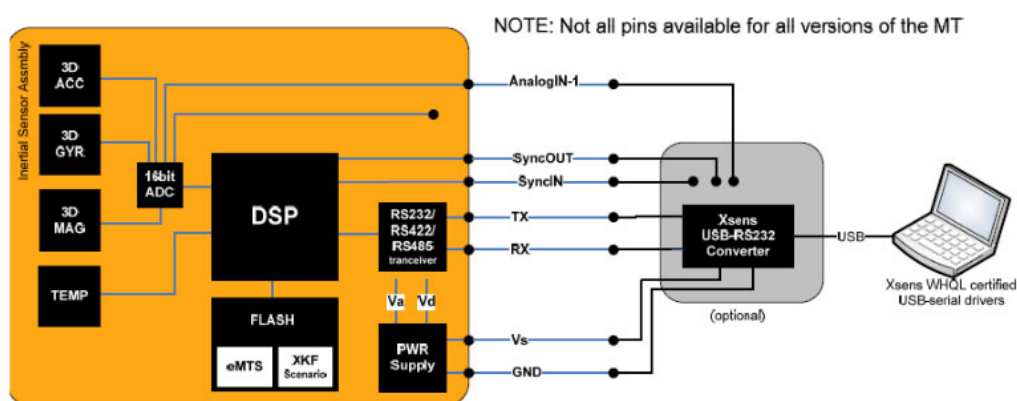


Figura 20. Esquema general del MTi-300.

En este trabajo solo se utilizarán los acelerómetros y los giroscopios. Los magnetómetros no se utilizarán debido a que no es importante estar posicionado en determinada dirección para la realización de los gestos.

Las características de los sensores de dicha IMU son las siguientes [35]:

Accelerometer	Magnitude	MTi-100 - series	
		Typical	Max
Standard full range	m/s ²	50	-
Bias repeatability (1 year)	m/s ²	0.03	0.05
In-run bias stability	μg	40	-
Bandwidth (-3 dB)	Hz	375	N/A
Noise density	μg /√Hz	80	150
Non-orthogonality	deg	0.05	0.05
Non-linearity	% FS	0.03	0.5
A/D resolution	bits	16	N/A

Tabla 2. Características de los acelerómetros del MTi-300.

Gyroscope	Magnitude	MTi-100 - series	
		Typical	Max
Standard full range	deg/s	450	-
Bias repeatability	deg/s	0.2	0.5
In-run bias stability	deg/h	10	-
Angular random walk	deg/h	-	-
Bandwidth (-3 dB)	Hz	450	-

Noise density	$\text{deg/s}/\sqrt{\text{Hz}}$	0.01	0.015
g-sensitivity (calibrated)	deg/s/g	0.003	0.015
Non-orthogonality	deg	0.05	-
A/D resolution	bits	16	N/A

Tabla 3. Características de los giróscopos del MTi-300.

El MTi se comunica a través de una línea serie USB que se conectará a un PC. El fabricante proporciona una API (comunicación a alto nivel) donde vienen definidas todas las funciones necesarias para su configuración así como para una correcta comunicación entre el PC y el MTi.

En caso de que el dispositivo se utilizara en algún tipo de sistema empotrado, se puede implementar una comunicación a más bajo nivel que no requiera la presencia de un PC. El protocolo de comunicación, que es el mensaje de base, permite al usuario ajustar los parámetros de la configuración del MTi y recuperar los datos de ésta. Así, se pueden modificar libremente (dentro de su rango de funcionamiento) los parámetros referentes a la frecuencia de muestreo, sincronización de salida, velocidad de transmisión, modo de salida de los datos, etc.

5.1. Estados de funcionamiento

El MTi cuenta con dos estados de funcionamiento (Figura 21):

- *Estado de configuración:* además de configurar el sensor como ya se ha comentado anteriormente, en este estado se obtienen los valores de la configuración actual. Los nuevos ajustes introducidos surtirán efecto cuando el dispositivo pase al estado de medición o cuando se vuelva a conectar, momento en el cual, el sensor arranca con la última configuración guardada.

- *Estado de medición:* en este estado el dispositivo carga la última configuración guardada y, en función de ésta, enviará los datos de salida al PC.

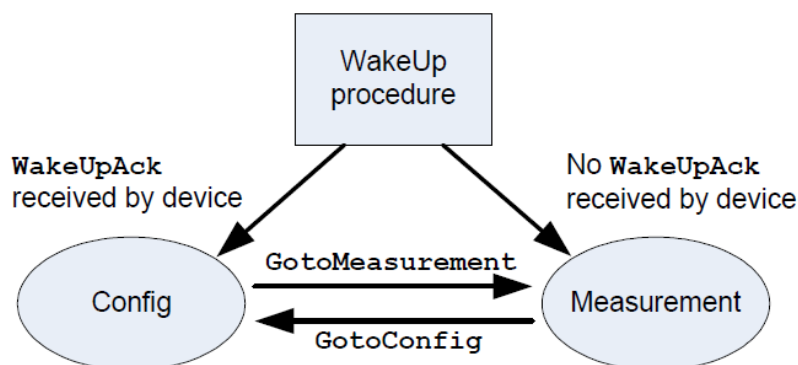


Figura 21. Estados del MTi-300.

Cualquier cambio realizado en la configuración del dispositivo debe hacerse siempre dentro del estado de configuración.

Existen dos formas de entrar en el estado de configuración o en el estado de medición. En el arranque o después de una señal de reset, el dispositivo empieza con el proceso 'WakeUp' (activación), si el controlador no responde al mensaje, entrará en el estado de medición con la última configuración almacenada. Pero si se envía el mensaje 'WakeUpAck' antes de 500ms desde la recepción del 'WakeUp' entrara en el estado de configuración. La otra forma de entrar en el estado de configuración o medición es usando los mensajes 'GoToConfig' o 'GoToMeasurement' mientras que esté el otro estado activo.

5.2. Comunicación a bajo nivel

La comunicación con el MTi se realiza a través de mensajes [36] cuya estructura estándar es la que refleja la siguiente figura

PREAMBLE	BID	MID	LEN	DATA	CHECKSUM
----------	-----	-----	-----	------	----------

Figura 22. Cuerpo del mensaje MTComm.

El número de datos que podrá contener cada mensaje tendrá una longitud de hasta 254 bytes. Existe un mensaje estándar extendido en el que la longitud máxima será de 2048 bytes. Los campos que componen los mensajes se muestran en la Tabla 4.

CAMPO	VALOR	DESCRIPCIÓN
PREAMBLE	1 byte	Cabecera, indicador de inicio de paquete →250 (0xFA).
BID	1 byte	Identificador del bus/dirección →255 (0xFF).
MID	1 byte	Identificador del mensaje
LEN	1 byte	Valor del número de bytes en el campo DATA. Valor máximo es 254 (0xFE). El valor 255 (0xFF) está reservado)
DATA	0 – 254 bytes	Bytes de datos (opcional).
CHECKSUM	1 byte	Mensaje de comprobación.

Tabla 4. Campos del mensaje MTComm.

5.3. Comunicación a alto nivel

Xsens proporciona un software de desarrollo (Software Development Kit, MT SDK) cuya misión principal es la de facilitar el desarrollo de aplicaciones específicas por parte del usuario basadas en el MTi-300.

El SDK consiste en un código fuente y una librería dinámica (.dll). El código fuente está disponible en C, ya que este lenguaje puede ser manejado por muchos otros lenguajes de programación como C++, Java y Python.

La Figura 23 representa una visión esquemática del MT SDK. Como se puede observar, el desarrollador de la aplicación host puede optar por utilizar una interfaz COM, C o C++. Sin embargo, sólo la interfaz de C se entrega como una librería dinámica. Para la interfaz de C++ el código fuente de las clases se suministran como parte del SDK.

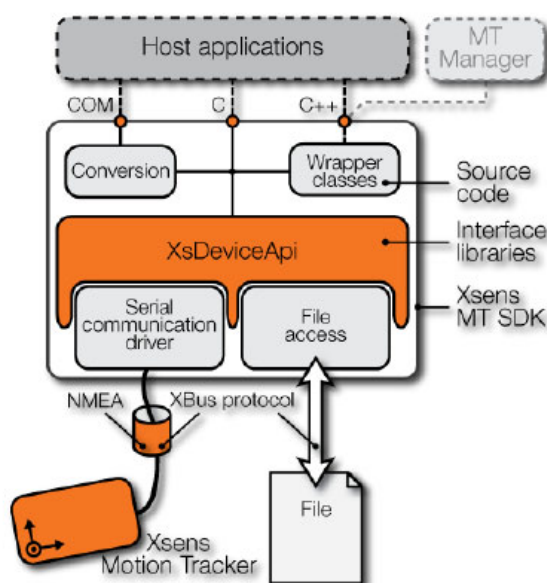


Figura 23. Xsens device API.

Será este tipo de comunicación a alto nivel, el empleado en este trabajo para la comunicación con el MTi-300. El SDK con sus librerías dinámicas serán instaladas en un PC y su utilización se realizará a través del programa Matlab.

Las funciones de control globales y de gestión del dispositivo se encuentran en el objeto XsControl. Para acceder a la funcionalidad de un dispositivo específico se utiliza el objeto XsDevice.

Los pasos generales a seguir para comenzar con la adquisición de datos de los sensores serán los siguientes:

1. Introducir el número de serie del dispositivo con `XsControl::setSerialKey`.
2. Escanear los dispositivos de Xsens con `XsScanner::scanPorts`.
3. Abrir el puerto de comunicación con `XsControl::openPort` y obtener el objeto del dispositivo `XsControl::device`.
4. Configurar el MTi con las funciones `XsDevice`.
5. Iniciar las medidas.

5.4. Salida de datos

Existen diversas formas en los que el MTi entrega los datos de los sensores:

- Modos de salida de la orientación:
 - Cuaterniones.
 - Ángulos de Euler.
 - Matriz de rotación.
- Datos de los sensores:
 - Calibrados.
 - Sin calibrar.

Se va a utilizar solamente la salida de los datos calibrados ya que a estos se les han corregidos errores tales como como los que aparecen en la siguiente tabla ([34]): Ganancia, offset, temperatura, desalineamiento...

Error Parameter	Symbol	Units
Roll Gyro Constant Offset	c_{ϕ}	$^{\circ}/\text{sec}$
Pitch Gyro Constant Offset	c_{θ}	$^{\circ}/\text{sec}$
Yaw Gyro Constant Offset	c_{ψ}	$^{\circ}/\text{sec}$
Roll Gyro Scale Factor Error	SF_{ϕ}	ppm
Pitch Gyro Scale Factor Error	SF_{θ}	ppm
Yaw Gyro Scale Factor Error	SF_{ψ}	ppm
Gyro Triad Misalignment about X	δ_{Gx}	μrad
Gyro Triad Misalignment about Y	δ_{Gy}	μrad
Gyro Triad Misalignment about Z	δ_{Gz}	μrad
Gyro Triad Nonorthogonality about X	Δ_{Gx}	μrad
Gyro Triad Nonorthogonality about Y	Δ_{Gy}	μrad
Gyro Triad Nonorthogonality about Z	Δ_{Gz}	μrad
Longitudinal Accelerometer Constant Offset	$c_{\bar{x}}$	g
Vertical Accelerometer Constant Offset	$c_{\bar{y}}$	g
Lateral Accelerometer Constant Offset	$c_{\bar{z}}$	g
Longitudinal Accelerometer Scale Factor Error	SF_x	ppm
Vertical Accelerometer Scale Factor Error	SF_y	ppm
Lateral Accelerometer Scale Factor Error	SF_z	ppm
Longitudinal Accelerometer Scale Factor Asymmetry	SFA_x	ppm
Vertical Accelerometer Scale Factor Asymmetry	SFA_y	ppm
Lateral Accelerometer Scale Factor Asymmetry	SFA_z	ppm
Longitudinal Accelerometer Scale Factor Nonlinearity	SFN_x	ppm
Vertical Accelerometer Scale Factor Nonlinearity	SFN_y	ppm
Lateral Accelerometer Scale Factor Nonlinearity	SFN_z	ppm
Accelerometer Triad Misalignment about X	δ_{Ax}	μrad
Accelerometer Triad Misalignment about Y	δ_{Ay}	μrad
Accelerometer Triad Misalignment about Z	δ_{Az}	μrad
Accelerometer Triad Nonorthogonality about X	Δ_{Ax}	μrad
Accelerometer Triad Nonorthogonality about Y	Δ_{Ay}	μrad
Accelerometer Triad Nonorthogonality about Z	Δ_{Az}	μrad

Tabla 5. Parámetros de error en una IMU.

6 Posicionamiento de la mano

6.1. Introducción

En el proceso de calcular la posición y la actitud de la mano se tiene que pasar por diferentes fases que a lo largo de este capítulo se irán explicando con detenimiento, desde cómo se obtienen los datos del sensor hasta que se ha realizado el cálculo de la posición y la actitud de la mano. Aquí, todavía no se realiza el reconocimiento del gesto que será el propósito del siguiente capítulo.

Como ya se comentó anteriormente, la comunicación con el MTi-300 se realiza a través de un puerto USB conectado a un PC y utilizando las librerías dinámicas que se proporcionan con el dispositivo.

Toda la comunicación y obtención de muestras se ha llevado a cabo a través de la programación de diversos códigos implementados en Matlab.

A lo largo de este capítulo, se hará referencia a varios archivos de Matlab (*.m) que contienen el código empleado en cada tarea y que estará incluido en el anexo de este libro.

6.2. Adquisición de los datos

Para la toma de muestras de los sensores se debe de configurar el MTi siguiendo los pasos que se describieron en el punto 5.3.

Se configurará el dispositivo para que entregue los datos calibrados de los tres acelerómetros y de los tres giróscopos que se almacenarán en un archivo. Estos datos serán una variable de tipo estructura llamada IMU (MTi_adquisicion_datos.m, apartado 11.1).

Los datos que se van a recoger serán los datos de salida calibrados de los sensores citados anteriormente. Se muestrearán los sensores a una frecuencia de 400 Hz y durante 3 segundos, tiempo suficiente para realizar cualquier gesto.

Como resultado de la adquisición de datos se puede observar en la Figura 24 las muestras obtenidas al realizar un movimiento con la mano hacia la derecha, izquierda y terminar en el punto donde se empezó.

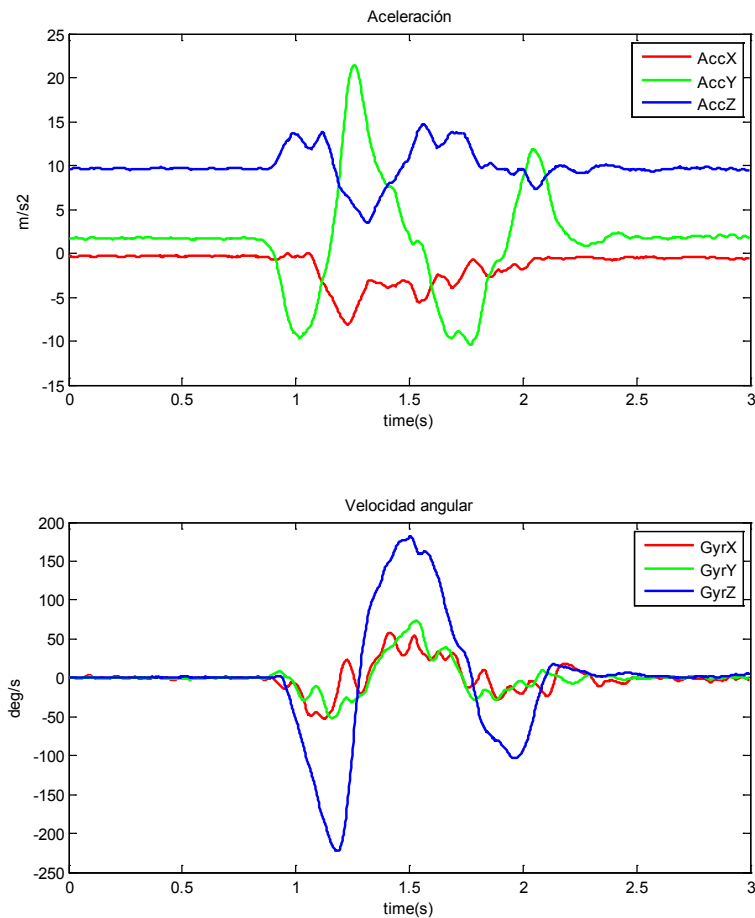


Figura 24. Resultado de la adquisición de datos.

6.3. Pulso en la mano

En la Figura 24 se mostraba las señales pertenecientes a los acelerómetros y los giróscopos al realizar un movimiento con la mano. En un primer momento, se puede creer que en las señales de los sensores existe cierto componente de ruido que hace que la señal tenga ciertas variaciones.

Ahora, vamos a detenernos en la Figura 25 a, en la que se han adquirido los datos del MTi cuando se encontraba colocado en el suelo y totalmente inmóvil. Se ve que prácticamente la señal es una línea recta y que no existe casi ninguna componente de ruido. En la Figura 25 b, se ha colocado la IMU en la mano y se ha intentado mantener lo más quieta posible. De nuevo, parece que la señal está contaminada por algún tipo de ruido.

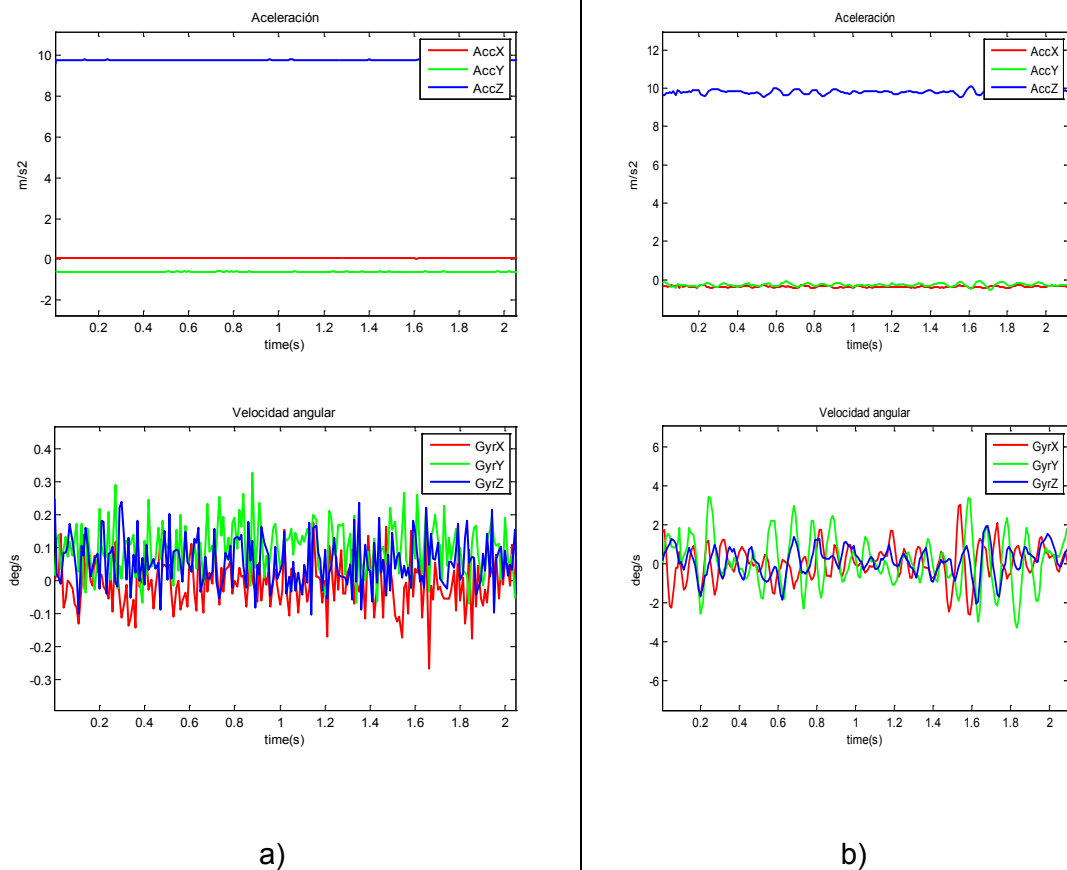


Figura 25. a) Señal del acelerómetro y giróscopo del eje X situado en una superficie plana. b) Señal de los mismos sensores situada en la mano en reposo.

Es importante destacar que las perturbaciones en la señal, en realidad, proceden de las vibraciones de la mano: las vibraciones producidas por el pulso de la persona se reflejan en la señal de los sensores.

Otro ejemplo lo tenemos en la Figura 26. En esta ocasión se ha realizado el gesto de un cuadrado moviendo la mano con cierta velocidad. Al finalizar el gesto se puede ver (círculo negro) como la mano tiene cierta tensión y se mueve más que al principio. Pasados unos segundos la mano se relaja y empiezan a desaparecer las vibraciones.

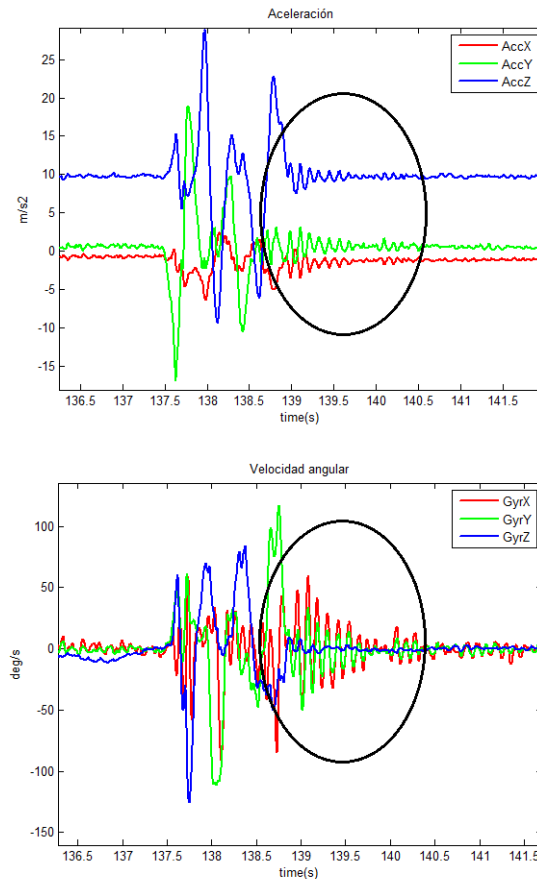


Figura 26. Temblor en la mano al realizar un gesto.

Por tanto, en los datos de los gestos existirá una componente de ruido más las vibraciones producidas por el temblor de la mano.

6.4. .Filtrado

Con lo visto en el punto anterior, cabría la posibilidad de implementar un filtro que eliminara el ruido presente en las señales. Pero, ¿a qué frecuencias?.

La frecuencia con la que puede vibrar la mano depende mucho de la persona: edad, cansancio, estados de ansiedad o nerviosismo... Un mismo individuo podría dar resultados bastante distintos a diversas horas del día.

Además, la frecuencia que pueda tener la señal del gesto puede ser parecida a la frecuencia de vibración de la mano, con lo que se podría perder información del movimiento realizado

Por tanto, lo que se ha hecho es un filtrado sencillo (filtro_media.m, apartado 11.4) que intente suavizar la señal lo máximo posible. El filtro utilizado es un filtro de medias que corresponde a la siguiente ecuación:

$$\bar{x}(n) = \frac{1}{k} \sum_{i=0}^k x(n+i)$$

Un dato muy importante a tener en cuenta es el número de muestras que se toman a la hora de la realización del filtrado. En este caso, se ha empleado una longitud de 12. Si la longitud es demasiado larga, la amplitud de la señal quedará demasiado reducida. Esta reducción de la señal para el caso de los acelerómetros tiene el siguiente efecto: si la aceleración ha disminuido, a la hora de calcular su posición, ésta también habrá disminuido, dando lugar a una posición errónea de la mano.

En la siguiente figura se observa el efecto del filtro en la señal de uno de los acelerómetros.

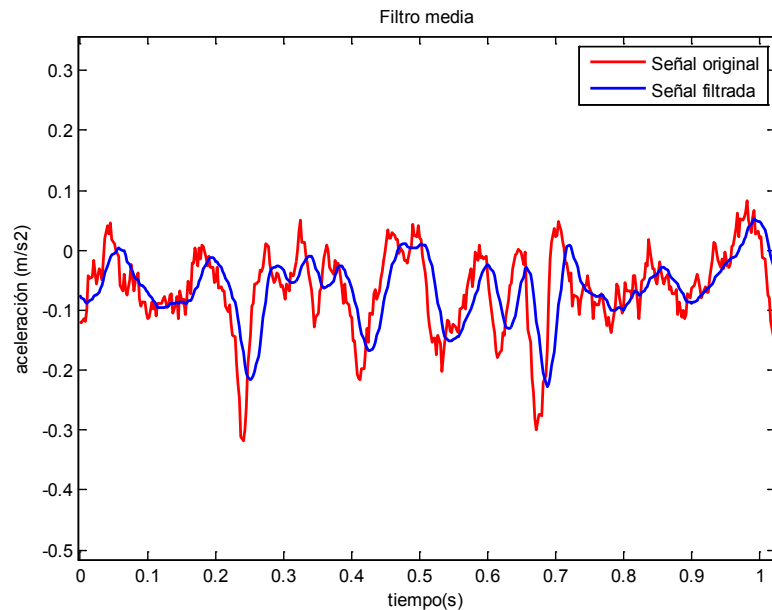


Figura 27 Señal del acelerómetro del eje X filtrada.

6.5. Segmentación

De toda la información que nos proporciona la IMU sólo nos interesa aquella que está comprendida entre el inicio y el fin del gesto realizado.

Para obtener dicha información es necesario segmentar de los datos adquiridos solo aquellos que sean de interés (segmentación_1eje.m, apartado 11.5).

La segmentación se realiza por comparación de niveles de la señal de los acelerómetros. Se establece un umbral y, si se sobrepasa, significa que la mano se está moviendo.

El funcionamiento para la segmentación de un solo eje es el siguiente:

1. Inicio del gesto. Se comprueba si la diferencia de aceleración entre la muestra k y la muestra $k+n$ es superior a un determinado nivel. Si es así, significa que ha habido algún tipo de movimiento, por lo que se toma como valor de inicio la muestra k . Si no, se sigue comprobando hasta que se supere dicho nivel. El umbral de inicio se establecerá en 0.7 m/s^2 .
2. Final del gesto. Ahora, se realiza lo mismo pero desde el final de los datos. Si la muestra $k-n$ es mayor en módulo a la muestra k , se toma como final del gesto la muestra k . El umbral final se establece en 1 m/s^2 .

Los umbrales de inicio y final poseen distintos valores debido a que al finalizar el movimiento de la mano y dejarla quieta, en ocasiones la mano está tensa y se notan más vibraciones. Esto ya se comentó en el apartado 6.3.

Analizando con detenimiento el funcionamiento descrito, cabe plantearse la posibilidad de qué pasaría si la señal no sobrepasase el nivel fijado como inicio o final del movimiento. De hecho, esta situación es común en casi todos los gestos analizados: cuando se mueve la mano a la derecha, idealmente, será el acelerómetro del eje Y el único que registra un cambio en su valor, estando la señal de los acelerómetros de los otros dos ejes libre de variaciones.

En la práctica, esto no ocurre tal y como se puede apreciar en la Figura 28 que representa el valor de los acelerómetros cuando se mueve la mano hacia la derecha y vuelve a su posición inicial. Claramente la señal del acelerómetro del eje y presenta

una variación importante mientras que en los otros dos acelerómetros también se ha registrado cierta variación. Como se ha comentado es muy difícil mover la mano siguiendo la dirección de un solo eje.

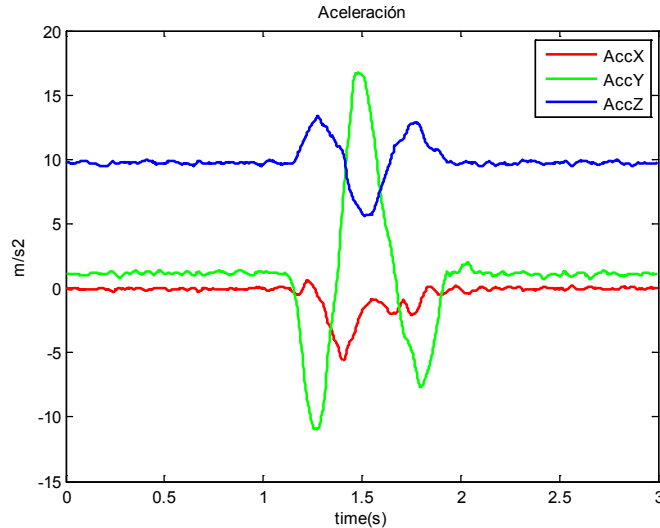


Figura 28. Señal de los acelerómetros cuando la mano se mueve a la derecha y vuelve al centro.

Sí es posible que si el gesto se ha realizado muy bien puede que el incremento en la señal en los otros ejes no sea lo suficientemente elevado como para sobrepasar el nivel estipulado adquiriendo la señal del acelerómetro una forma de casi línea recta. Cuando ocurre esto, lo que se hace es que el inicio se establece en la mitad de las muestras y, el final, en la mitad más uno.

Una vez se han obtenido el valor inicial y final de cada uno de los ejes, los resultados indican que cada eje tiene un valor distinto. Para tomar la decisión de cuáles son los valores correctos de inicio y fin se realiza lo siguiente:

- El valor de inicio del gesto será el mínimo de los tres valores.
- El valor de fin será el máximo de los tres valores.

Siguiendo con el ejemplo del movimiento de la mano hacia la derecha, el inicio de los gestos en cada uno de los tres ejes será el siguiente:

- Inicio en acelerómetro eje x: 1.198 s.
- Inicio en acelerómetro eje y: 1.08 s.
- Inicio en acelerómetro eje z: 1.078 s.

Y el final de los mismos será:

- Final en acelerómetro eje x: 1.89 s
- Final en acelerómetro eje y: 1.987 s
- Final en acelerómetro eje z: 1.995 s

Por tanto, si el inicio es el menor de los tres y el final el mayor de estos, el gesto quedará delimitado entre los siguientes tiempos (zona central de la Figura 29):

- Inicio del gesto: 1.89 s
- Final del gesto: 1.987 s

La segmentación de los tres eje se encuentra en `segmentación_3ejes.m` en el apartado 11.6.

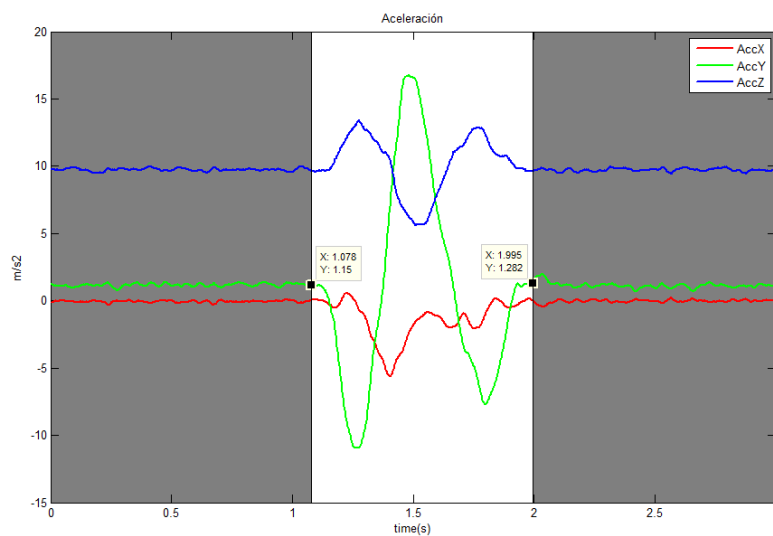


Figura 29. Segmentación de la señal de los acelerómetros.

Para finalizar, se recortará todo el conjunto de los datos, tanto los datos de los acelerómetros como de los giróscopos, con el mismo inicio y el mismo fin en cada uno de ellos.

6.6. Obtención del ángulo inicial

La orientación de la mano quedará definida por los valores de roll, pitch y yaw. Pero lo primero que se debe saber es cual la orientación inicial.

Dicho valor, se obtendrá a partir de los acelerómetros.

Con este valor de aceleración y, utilizando los acelerómetros como inclinómetros se puede obtener el ángulo de cada uno de los ejes con las siguientes ecuaciones [33]:

$$\text{ángulo eje } x \quad \phi = \tan^{-1} \left(\frac{A_{X,OUT}}{\sqrt{A_{Y,OUT}^2 + A_{Z,OUT}^2}} \right)$$

$$\text{ángulo eje } y \quad \theta = \tan^{-1} \left(\frac{A_{Y,OUT}}{\sqrt{A_{X,OUT}^2 + A_{Z,OUT}^2}} \right)$$

$$\text{ángulo eje } z \quad \psi = \tan^{-1} \left(\frac{A_{Z,OUT}}{\sqrt{A_{X,OUT}^2 + A_{Y,OUT}^2}} \right)$$

Las anteriores ecuaciones indican qué ángulo tiene cada eje de los acelerómetros con respecto de la gravedad.

El valor del ángulo en el eje x corresponde al pitch y el del eje y, al roll. El valor inicial de yaw siempre lo supondremos igual 0. Esto es debido a que la orientación en este ángulo no es de importancia cuando comienza el gesto, es decir, no es importante hacia qué zona del horizonte se encuentra mirando la mano.

Hacer notar que el código de este bloque y el de los sucesivos hasta el punto 6.12 se encuentra en posicionamiento.m (apartado 11.7)

6.7. Actitud

Una vez obtenido la actitud inicial de la mano y con las medidas de los giróscopos se procede a calcular su actitud como describe *George M. Siouris* en [32].

Las componentes de cada una de las derivadas de los ángulos de Euler se deben añadir vectorialmente a lo largo de un eje dado del sistema de coordenadas de *body* a los componentes de ω en ese sistema:

$$\omega = \dot{\psi} + \dot{\theta} + \dot{\phi}$$

Así pues, se pueden expresar las velocidades angulares de cada uno de los giróscopos de la siguiente manera:

$$\omega_x = p = \dot{\psi}_x + \dot{\theta}_x + \dot{\phi}_x = \dot{\phi} - \dot{\psi} \sin \theta$$

$$\omega_y = q = \dot{\psi}_y + \dot{\theta}_y + \dot{\phi}_y = \dot{\psi} \cos \theta \sin \phi + \dot{\theta} \cos \phi$$

$$\omega_z = r = \dot{\psi}_z + \dot{\theta}_z + \dot{\phi}_z = \dot{\psi} \cos \theta \cos \phi - \dot{\theta} \sin \phi$$

Para expresar ψ , θ , and ϕ en términos de (p, q, r) se resuelven las ecuaciones anteriores utilizando determinantes:

$$\dot{\psi} = (1/\cos \theta)(q \sin \phi + r \cos \phi)$$

$$\dot{\theta} = q \cos \phi - r \sin \phi$$

$$\dot{\phi} = p + \tan \theta (q \sin \phi + r \cos \phi)$$

Las ecuaciones anteriores relacionan los componentes de la velocidad angular del sistema de referencia de la mano con la tasa de cambio de los ángulos de Euler pudiendo expresarse del siguiente modo:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin \phi \tan \theta & \cos \phi \tan \theta \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi \sec \theta & \cos \phi \sec \theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

A través de esta última ecuación se puede calcular la actitud que tiene la mano en todo momento.

Sin embargo, se puede observar en la última ecuación que aparece un problema a la hora de expresar la orientación en términos de los ángulos de Euler. Por ejemplo, si el pitch se aproxima a $\pm 90^\circ$ aparece una singularidad en la ecuación por lo que el resultado no sería fiable. A este problema se le conoce como *gimbal-lock*.

En los gestos que se van a intentar reconocer, no se debería de alcanzar ese valor, por lo que en este trabajo no se presentaría dicho problema.

La forma de solucionarlo sería expresar los giros en forma de cuaterniones, evitando así posibles indeterminaciones.

Aunque, ahora no se dé la circunstancia del *gimbal-lock*, si en un futuro se desean reconocer otros gestos en los que sí se pueda dar dicho problema, en el apartado siguiente se explica la forma de expresar la orientación en cuaterniones.

6.7.1. Sistema de cuaterniones

Un cuaternión está formado por una cuádrupla de números reales, escritos en un orden definido como un vector tridimensional.

$$[Q] = q_0 + q_1i + q_2j + q_3k = (q_0, q_1, q_2, q_3) = (q, q)$$

Los elementos del cuaternión q_1, q_2, q_3 describen un vector en el espacio, mientras que q_0 , indica la cantidad que se rota respecto a ese vector.

$$Q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos \frac{\mu}{2} \\ \sin \frac{\mu}{2} \cos \alpha \\ \sin \frac{\mu}{2} \cos \beta \\ \sin \frac{\mu}{2} \cos \gamma \end{bmatrix}$$

Con estos cambios, se consigue pasar de un sistema de tres ángulos (Euler) a los cuatro parámetros de los cuaterniones, que definen de igual manera la actitud de un cuerpo.

La correspondencia entre la matriz dirección coseno y los cuaterniones es:

$$C = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1q_2 - q_3q_0) & 2(q_3q_1 + q_2q_0) \\ 2(q_3q_0 + q_1q_2) & 1 - 2(q_3^2 + q_1^2) & 2(q_2q_3 - q_0q_1) \\ 2(q_3q_0 - q_2q_1) & 2(q_2q_3 + q_0q_1) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix}$$

Y entre los componentes del cuaternión y los ángulos de Euler:

$$q_0 = \cos \frac{\psi}{2} \cos \frac{\theta}{2} \cos \frac{\phi}{2} - \sin \frac{\psi}{2} \sin \frac{\theta}{2} \sin \frac{\phi}{2}$$

$$q_1 = \sin \frac{\theta}{2} \sin \frac{\phi}{2} \cos \frac{\psi}{2} + \sin \frac{\psi}{2} \cos \frac{\theta}{2} \cos \frac{\phi}{2}$$

$$q_2 = \sin \frac{\theta}{2} \cos \frac{\psi}{2} \cos \frac{\phi}{2} - \sin \frac{\psi}{2} \sin \frac{\phi}{2} \cos \frac{\theta}{2}$$

$$q_3 = \sin \frac{\phi}{2} \cos \frac{\psi}{2} \cos \frac{\theta}{2} + \sin \frac{\psi}{2} \sin \frac{\theta}{2} \cos \frac{\phi}{2}$$

Con estas ecuaciones de relación, se puede pasar del sistema de ángulos de Euler, incluso de la matriz DCM a cuaterniones, resolver la matriz evitando así el problema del *gimbal-lock* y volviendo a transformar.

Esta es la matriz que relaciona las velocidades angulares, equivalente a la ecuación planteada en el apartado anterior.

$$\begin{bmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix} \begin{bmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{bmatrix}$$

Con estos cambios se eliminan las singularidades.

6.8. Eliminación de la gravedad

Los acelerómetros además de medir la aceleración lineal también son sensibles a la aceleración de la gravedad. Si se coloca uno de forma perpendicular al vector de la gravedad el valor de su salida, de forma general, será de 9.80665 m/s^2 . Con este valor, si intentamos calcular su posición integrando dos veces dicho valor, después de unos segundos el resultado sería que se ha avanzado muchos metros mientras que el sensor ni se ha movido.

Intentar eliminar el error en la medición es una de las tareas más importantes cuando se habla de posicionamiento o navegación.

En las coordenadas de navegación la aceleración debida a la gravedad en cada uno de los tres ejes sería $[0 \ 0 \ g]$, tomando como g el valor genérico de la gravedad 9.80665 m/s^2 .

Pero lo que interesa es quitar esta componente en las aceleraciones medidas por el sensor, es decir, las coordenadas de *body*.

Por tanto, debemos pasar de coordenadas de navegación a *body* a través de la matriz de cambio de coordenadas C_b^n . La siguiente ecuación nos proporciona el valor de la aceleración de la gravedad en cada uno de los ejes:

$$g_b = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} = C_b^n \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} -g \sin\theta \\ g \cos\theta \sin\phi \\ g \cos\theta \cos\phi \end{bmatrix}$$

Ahora ya solo falta restar la componente de la gravedad del valor de aceleración que proporcionan los acelerómetros.

Por ejemplo, se va a realizar un giro de la mano hacia la derecha, en sentido de las agujas del reloj (Figura 30). Con la mano quieta, los ejes x e y están prácticamente perpendiculares a la gravedad, por lo que el valor de la aceleración en los acelerómetros de dichos ejes es casi nulo. En el eje Z , que está paralelo a la gravedad pero en sentido contrario, el valor de la aceleración es de $-g$, es decir, aproximadamente -9.80665 m/s^2 .

Cuando giramos la mano hacia la derecha los ejes afectados por la gravedad serán el z y el y: el primero tenderá a 0 y el segundo a 1 g. En el eje x se observa alguna variación ya que es imposible girar la mano sin moverla en los tres ejes.

La curva de color verde es el resultado del cálculo de g_b gracias a la información aportada de los giróscopos y al cálculo realizado de los ángulos de orientación. Si se elimina el efecto de la gravedad de la señal de aceleración, se puede observar como la aceleración resultante es casi nula.

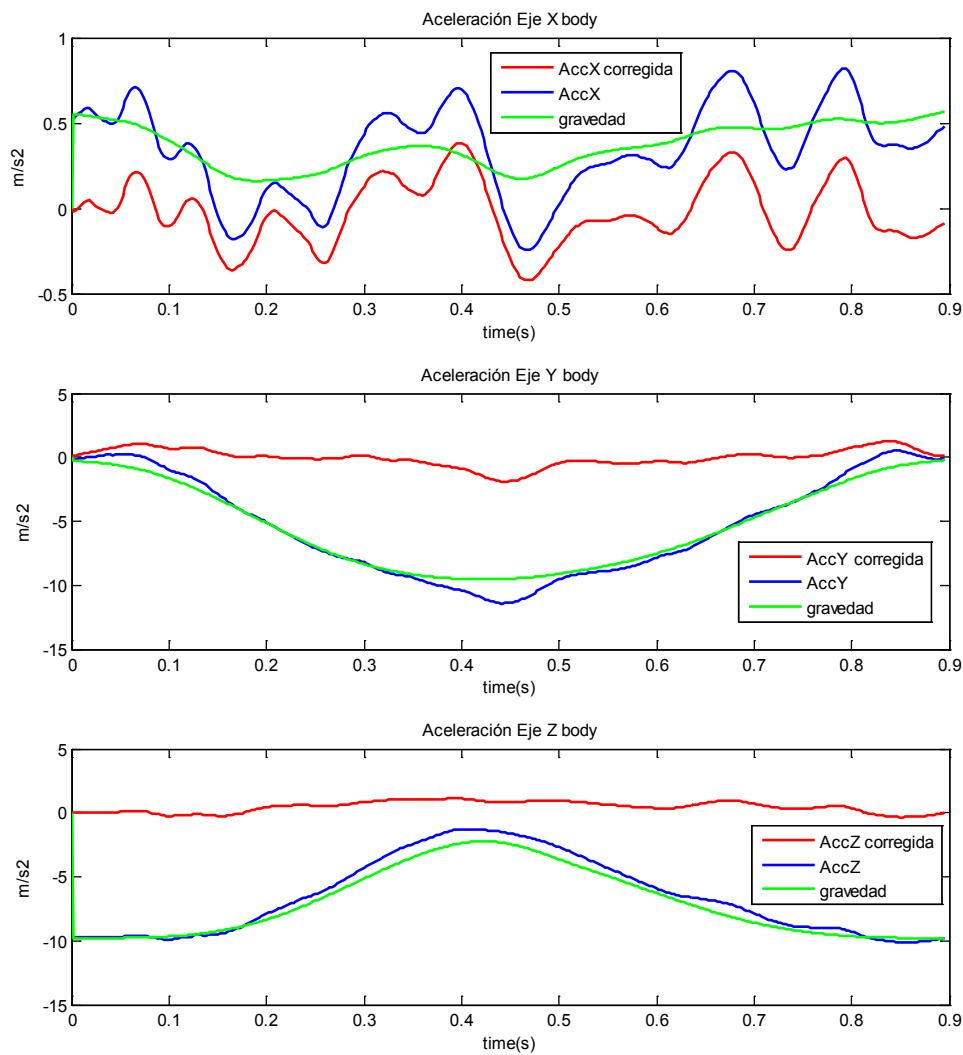


Figura 30. Eliminación de la gravedad en los acelerómetros.

6.9. Cambio de coordenadas

Una vez se ha obtenido el valor de las aceleraciones sin el efecto de la gravedad en el marco de *body*, ahora ya solo falta transformarlas a coordenadas de navegación.

Se utiliza la matriz de cambio coordenadas de la siguiente forma:

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}^n = C_b^n \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}^b$$

En la siguiente figura se puede apreciar el valor del cambio de coordenadas de la aceleración en el cual se ha realizado un movimiento hacia la derecha con la mano girada aproximadamente unos 45° en sentido de las agujas del reloj.

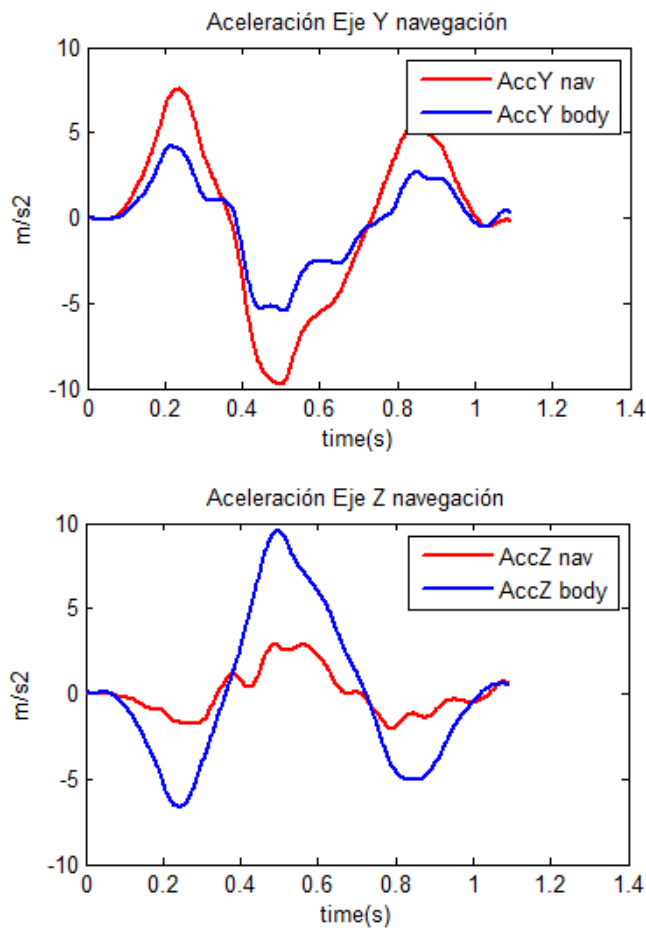


Figura 31. Aceleración en coordenadas de *body* y navegación.

Por tanto, se registra en las coordenadas de *body* cierta aceleración tanto en el eje z como en el eje y. Entonces al calcular la posición integrando dos veces la aceleración el resultado sería que la mano se ha movido en los dos ejes y sería como si se hubiese realizado una diagonal. Pero realmente la mano solo se ha movido hacia la derecha.

Cuando se realiza el cambio de coordenadas se aprecia en la figura anterior como se atenúa la aceleración en el eje z y como aumenta en el eje y. Por tanto, al calcular la posición se obtendría que la distancia en el eje y es considerable (aproximadamente unos 26 cm) mientras que en el eje z se habría movido unos pocos centímetros (alrededor de 5 cm).

6.10. Corrección de velocidad

Obtenida la aceleración en coordenadas de navegación, se integra ésta con el fin de obtener los valores de velocidad.

La integral realizada responde a la siguiente expresión:

$$v[n] = \int a \cdot dt \approx v[n - 1] + a[n] \cdot \Delta t$$

El resultado obtenido de calcular la velocidad al gesto de mover la mano hacia adelante es la Figura 32.

La figura anterior revela que el valor de velocidad final no es cero. Esto puede ser debido a varias razones:

- Deriva en el giroscopio.
- Error en la corrección de la gravedad.
- Errores en la integral.

Estos errores se irán acumulando en el tiempo por lo que se verán reflejados a la hora de realizar el cálculo de la posición.

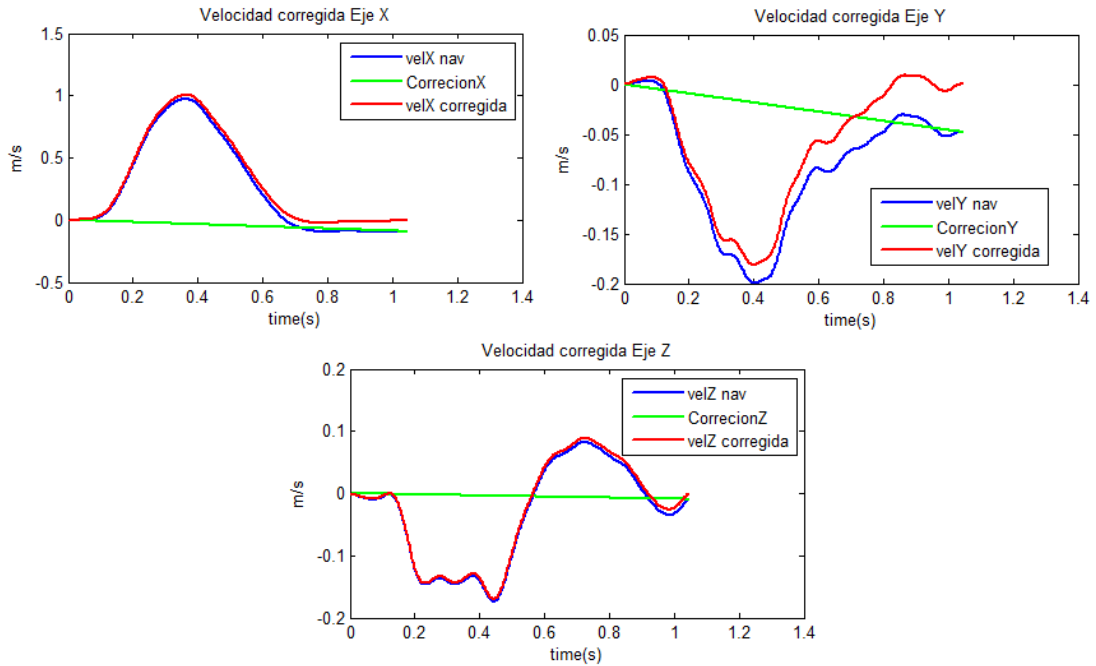


Figura 32. Velocidad corregida para el gesto adelante.

Se pueden evitar estos errores en la velocidad mediante la siguiente suposición:

“Todo gesto realizado debe empezar y acabar con una velocidad igual a cero”

Todo valor en la aceleración distinto de cero, al integrarla, se verá reflejado en la velocidad de una forma lineal. A este tipo de corrección se la conoce como “Zero Velocity Compensation” o ZVC ([40], [52]).

Tomando el valor inicial y el final de la velocidad se puede calcular el valor del error para cada muestra (recta de color verde en la Figura 32) a partir de la ecuación de una recta:

$$\text{Corrección velocidad}[n] = nT_s \cdot \frac{\text{Velocidad final}}{\text{tiempo final}}$$

Si realizamos la corrección a la velocidad de la Figura 32 el resultado es el la señal de aceleración de color rojo.

6.11. Posición de la mano

El siguiente paso será calcular la posición de la mano a partir de los datos de velocidad que proporciona el bloque anterior.

$$p[n] = \int v \cdot dt \approx p[n-1] + v[n] \cdot \Delta t$$

Siguiendo con el ejemplo del apartado anterior en el que se había realizado un movimiento de la mano hacia adelante, si se integra la señal de la velocidad se obtiene la posición de la mano:

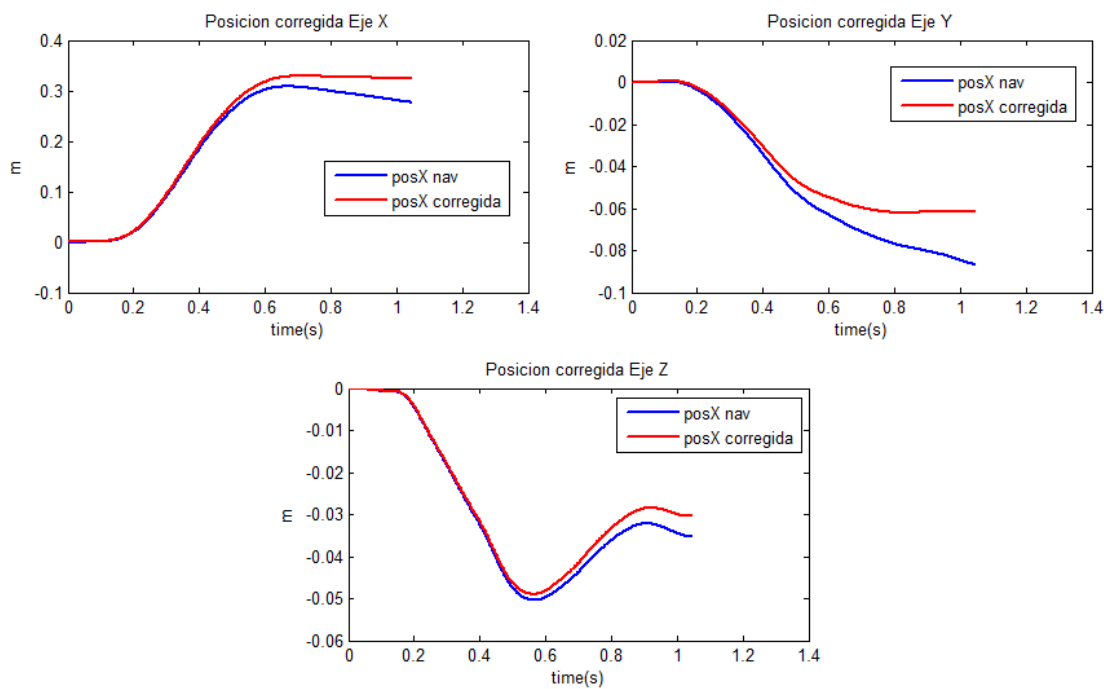


Figura 33. Posición de la mano en gesto adelante.

En la figura anterior se muestra la posición obtenida sin corrección de velocidad (azul) y con la corrección de velocidad realizada (rojo).

6.12. Cálculo de la distancia

Mientras se van procesando todas las muestras y se calcula la posición, se va comparando los valores de las posiciones y los ángulos de pitch y roll con el objetivo de obtener su valor máximo y mínimo. Por tanto, los valores que se obtienen son:

- Valor máximo del eje x.
- Valor mínimo del eje x.
- Valor máximo del eje y.
- Valor mínimo del eje y.
- Valor máximo del eje z.
- Valor mínimo del eje z.
- Valor máximo del pitch.
- Valor mínimo del pitch.
- Valor máximo del roll.
- Valor mínimo del roll.
- Valor de la última posición del eje x.
- Valor de la última posición del eje y.
- Valor de la última posición del eje z.

Los valores anteriores serán las entradas al bloque de lógica difusa que será el encargado de reconocer los gestos realizados.

7 Reconocimiento por lógica difusa

Una vez calculada la posición de la mano durante la realización del gesto y con los datos obtenidos en el apartado 6.12 ahora es el turno de reconocer qué gesto ha realizado una persona.

Se ha utilizado la herramienta *FIS editor* de Matlab y que en la Figura 34 se puede ver como se ha realizado el bloque difuso.

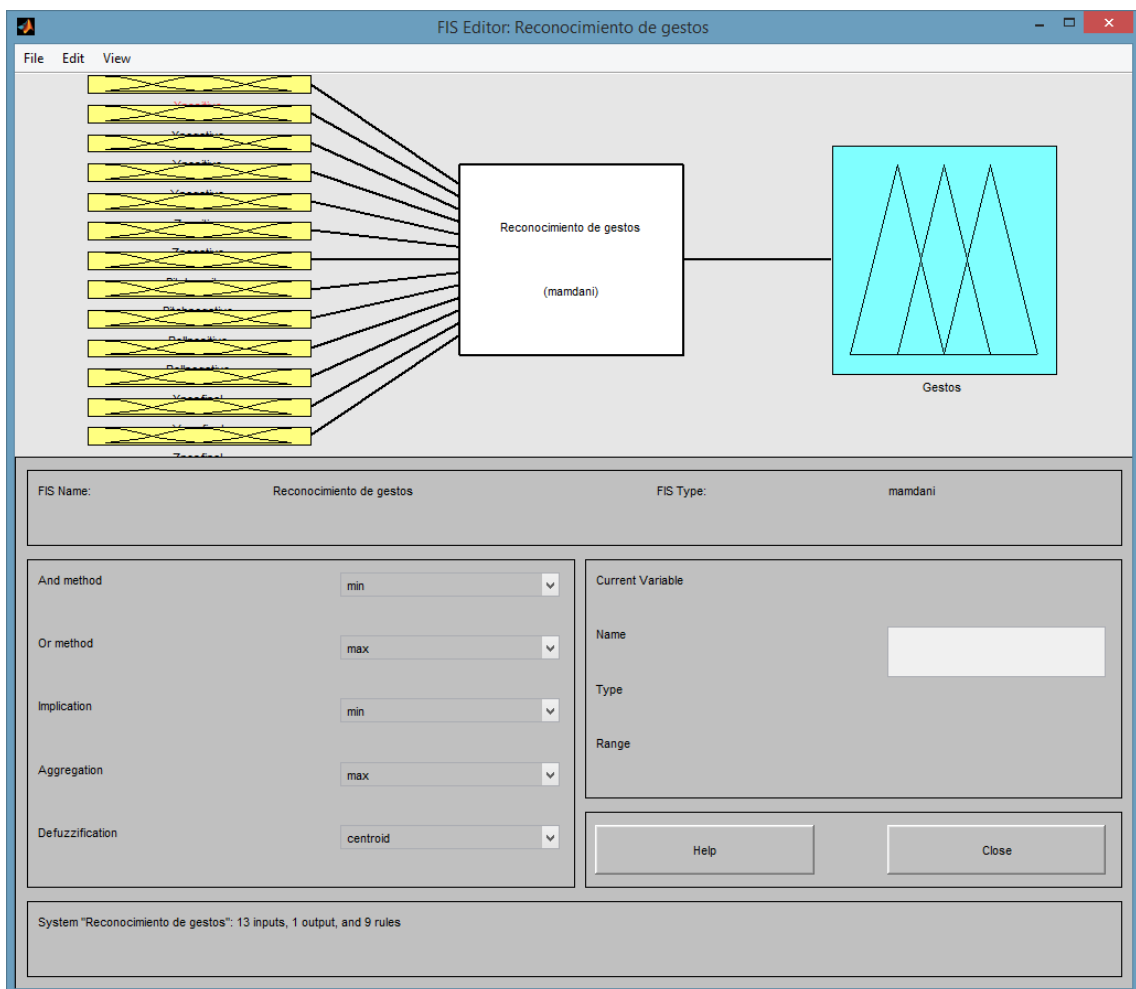


Figura 34. Bloque de lógica difusa realizado con FIS editor.

7.1. Variables de entrada

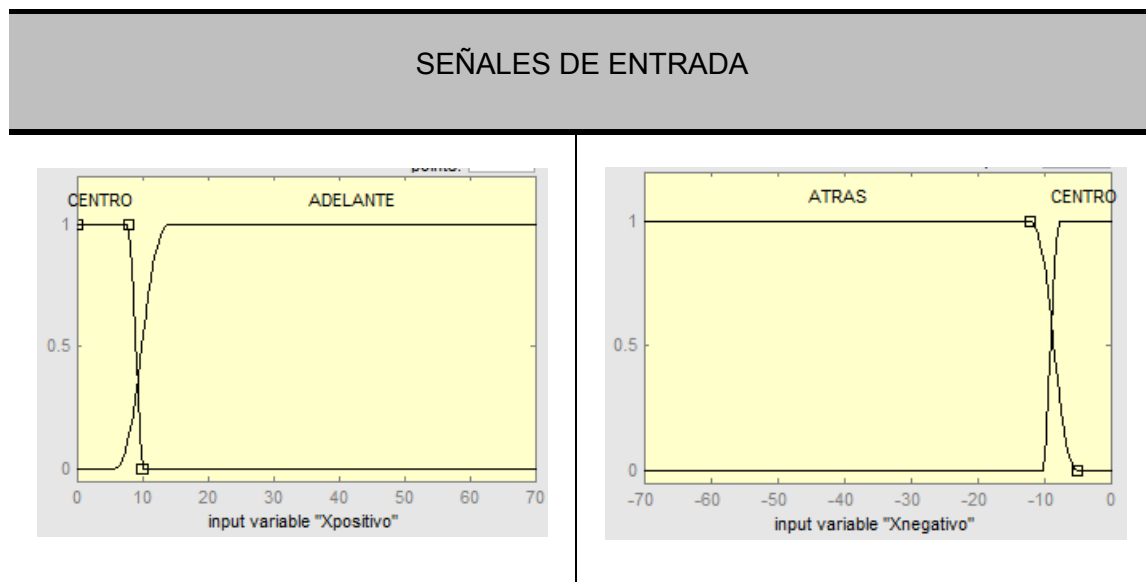
En el apartado 6.12 ya se describieron las variables que van a ser la entrada al conjunto reconocedor de gestos basado en lógica difusa.

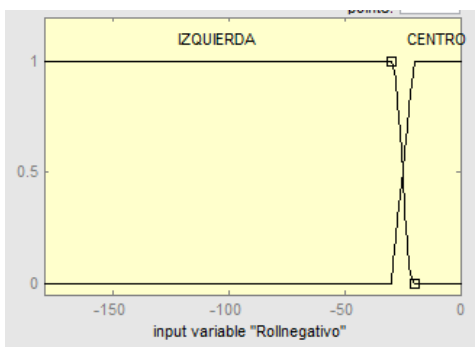
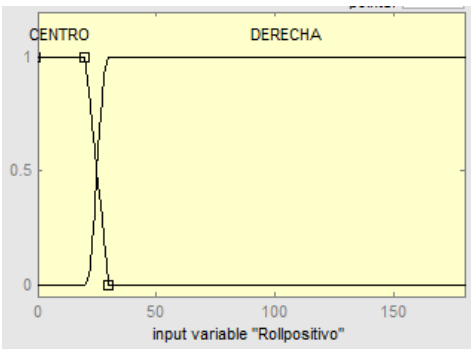
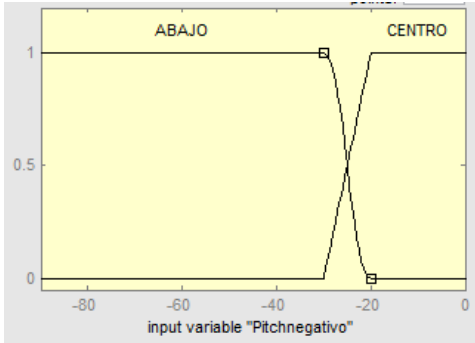
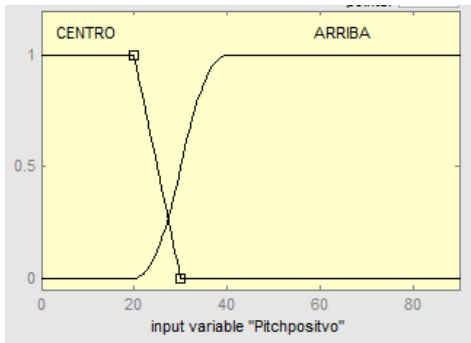
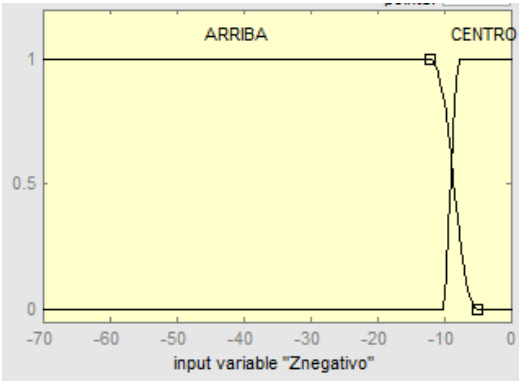
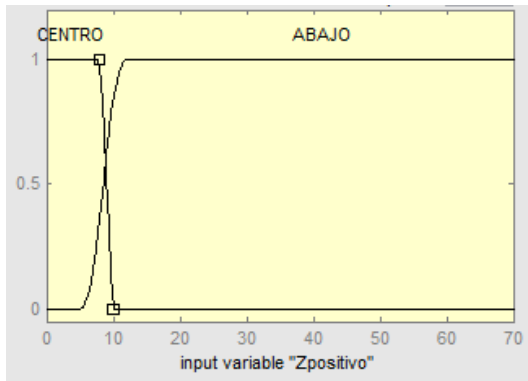
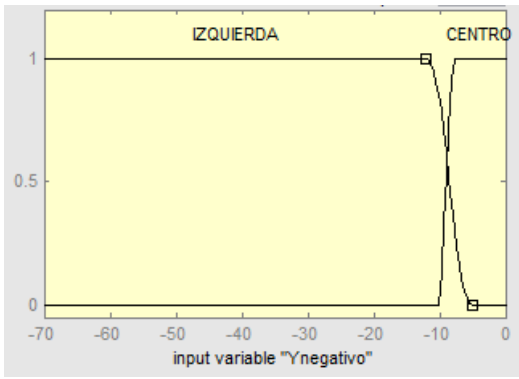
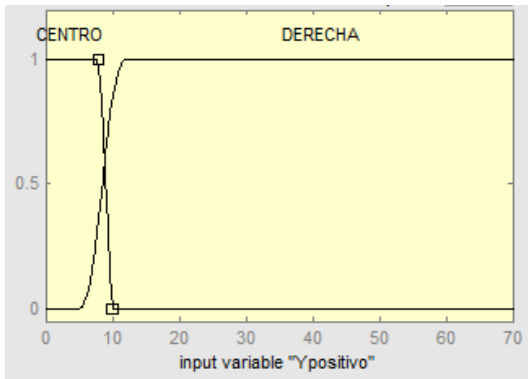
El universo de discurso se situará para todos los ejes en 70 cm en caso de las zonas positivas de los mismos y de -70 cm para las negativas. En principio, con los gestos propuestos, lo normal es que estos valores sean la distancia máxima que alcance la mano.

En el caso del pitch, se situará en 90° y -90° para valores positivos y negativos respectivamente. Aunque recordando lo que se expuso en el apartado 6.7 en estos valores aparece el efecto del *gimbal-lock*.

Para el ángulo de roll su universo de discurso quedará definido entre 180° para valores positivos del ángulo y -180° para valores negativos.

En la siguiente tabla se pueden ver todas las variables de entrada del sistema de lógica difusa con sus respectivas funciones de pertenencia.





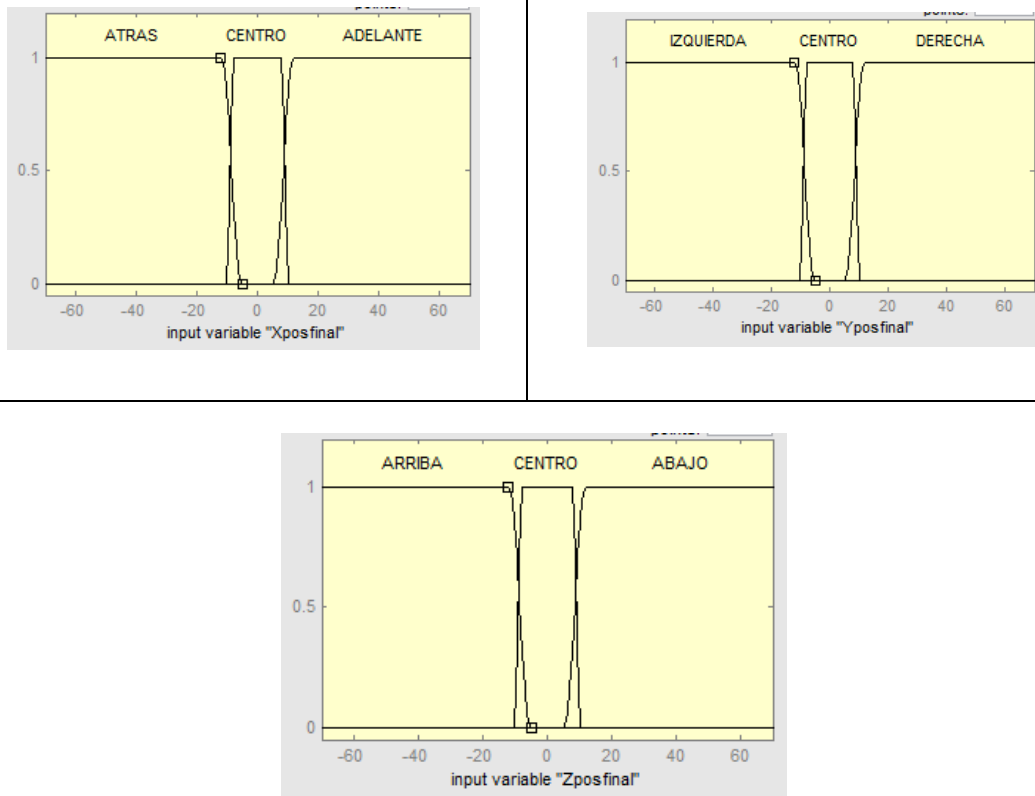


Tabla 6. Variables de entrada en el bloque de lógica difusa.

7.2. Reglas de inferencia

En la Tabla 7 se muestran las reglas de inferencia que se han utilizado con el fin de reconocer los gestos.

Las entradas a este bloque son conjuntos difusos (grados de pertenencia) y las salidas son también conjuntos difusos, asociados a la única variable de salida.

Se podrían añadir más gestos añadiendo más reglas y con las mismas entradas. En este trabajo solo se reconocerán 9 gestos por lo que se han desarrollado nueve reglas.

		ENTRADAS												
		Xpos	Xneg	Ypos	YNeg	Zpos	Zneg	Pitchpos	Pitchneg	Rollpos	Rollneg	Xposfinal	Yposfinal	Zposfinal
SALIDAS	DERECHA	Centro	Centro	Derecha	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Derecha	Centro
	ADELANTE	Adelante	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Adelante	Centro	Centro
	DERECHAcen	Centro	Centro	Derecha	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro
	IZDAcent	Centro	Centro	Centro	Izquierda	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro
	DERC-IZDA	Centro	Centro	Derecha	Izquierda	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro
	G.DCHA	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Derecha	Centro	Centro	Centro	Centro
	G.IZDA	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Centro	Izquierda	Centro	Centro	Centro
	CUADRADO	Centro	Centro	Derecha	Centro	Centro	Arriba	Centro	Centro	Centro	Centro	Centro	Centro	Centro
	ONDA	Centro	Centro	Derecha	Centro	Abajo	Arriba	Centro	Centro	Centro	Centro	Centro	Derecha	Centro

Tabla 7. Reglas de inferencia.

7.3. Salida

El sistema de lógica difusa creado, solamente tendrá una salida cuyo valor nos indicará el gesto realizado. En la siguiente figura se muestran sus funciones de pertenencia.

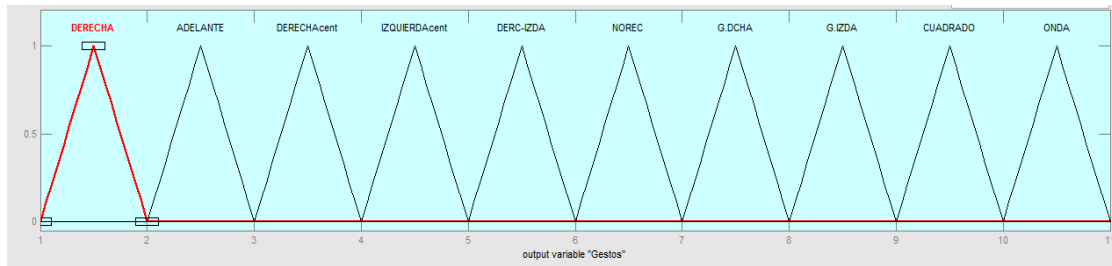


Figura 35. Variable de salida del sistema de lógica difusa.

Cada una de las funciones de pertenencia corresponde a cada uno de los gestos que se intentan identificar exceptuando una. Esta última, indicará que el gesto no se ha reconocido.

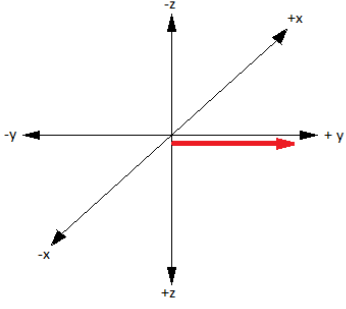
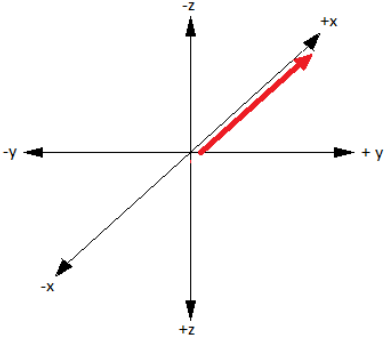
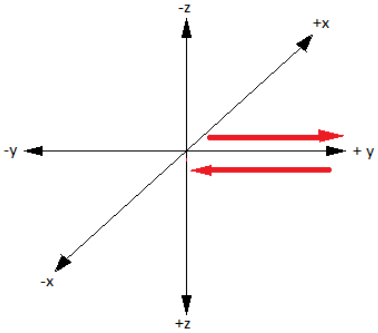
Como se puede comprobar, no es una salida normal para un sistema de lógica difusa ya que sus funciones de pertenencia no están entrelazadas.

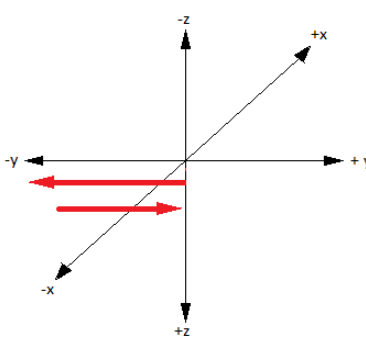
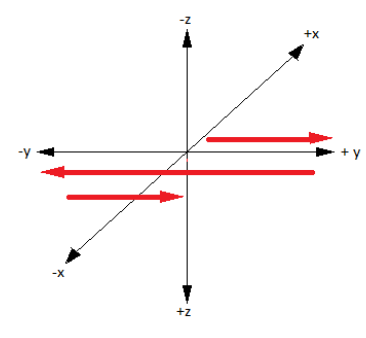
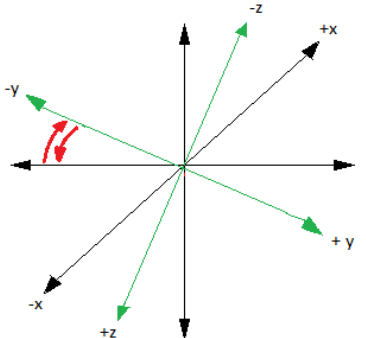
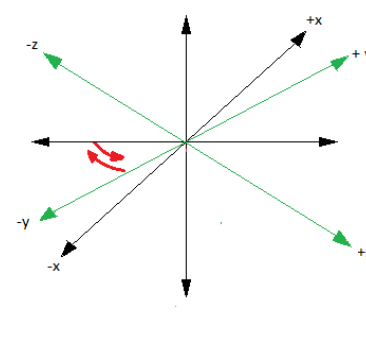
Pero para la aplicación del reconocimiento de gestos se ha adaptado el sistema difuso a las necesidades del proyecto dando bastantes buenos resultados como se podrá comprobar en el siguiente apartado.

8 Pruebas y resultados

8.1. Pruebas realizadas

Para comprobar el correcto funcionamiento del reconocedor de gestos se han adquirido 50 muestras de todos los gestos tal como muestra la siguiente tabla:

Gesto	Gráfica
Derecha	 <p>A 3D coordinate system with axes labeled $+x$, $-x$, $+y$, $-y$, $+z$, and $-z$. A single red arrow originates from the origin and points along the positive y-axis.</p>
Adelante	 <p>A 3D coordinate system with axes labeled $+x$, $-x$, $+y$, $-y$, $+z$, and $-z$. A single red arrow originates from the origin and points along the positive x-axis.</p>
Derecha - Centro	 <p>A 3D coordinate system with axes labeled $+x$, $-x$, $+y$, $-y$, $+z$, and $-z$. Two red arrows originate from the origin and point along the positive y-axis in opposite directions, one towards $+y$ and one towards $-y$.</p>

Izquierda - Centro	
Derecha - Izquierda	
Giro derecha	
Giro izquierda	

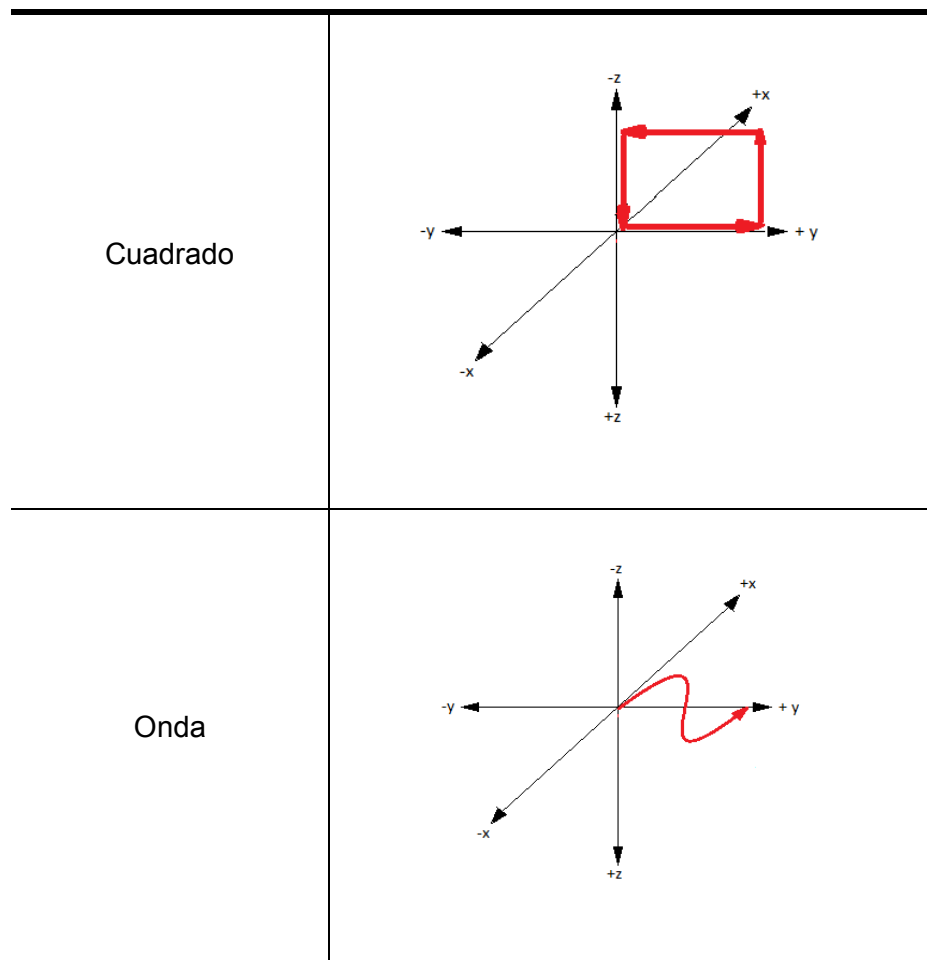


Tabla 8. Descripción de los gestos manuales.

En la toma de muestras han participado un total de 5 personas con edades comprendidas entre los 32 y los 62 años. Cada una de ellas ha realizado cada gesto un total de 10 veces por lo que el número total de muestras tomadas a cada persona es de 90 y haciendo un total de 450.

No se han tenido en cuenta factores tales como la hora del día, si se había realizado algún tipo de esfuerzo físico con anterioridad a la toma de muestras, etc. Por lo que la colección de datos se ha realizado de forma aleatoria en cuanto a la situación y condición física de los participantes.

Las muestras tienen una duración de 3 segundos cada una, tiempo más que suficiente para la realización de cualquiera de los gestos propuestos. Como ya se explicó en apartados anteriores, se segmentará del total de la muestra la información perteneciente exclusivamente al gesto, descartando lo demás.



Figura 36. Posición inicial.

La posición inicial para la realización de los gestos es la que se puede ver en la figura anterior.

8.2. Resultados

Una vez obtenidos un gran número de muestras, tal y como se describía en el apartado anterior, se comprueba el funcionamiento del reconocedor de gestos.

En las tablas siguientes se muestra el porcentaje de acierto y de fallo del reconocedor. Dentro de los fallos se califican en dos tipos:

- Los que han fallado reconociendo un gesto y lo clasifican como no reconocido.
- Los que se han confundido de gesto: habiendo realizado un movimiento a la izquierda, por ejemplo, lo confunden con un giro arriba o cualquier otro gesto.

Los resultados son los siguientes:

Gesto	Nº muestras	Correctas	Incorrectas mismo gesto	Incorrectas gesto equivocado	% Reconocimiento
DERECHA	50	50	0	0	100 %
ADELANTE	50	50	0	0	100 %
DERECHA - CENTRO	50	48	2	0	96 %
IZQUIERDA - CENTRO	50	48	2	0	96 %
DERECHA - IZQUIERDA	50	45	5	0	90 %

Tabla 9. Resultado de las pruebas (I).

Gesto	Nº muestras	Correctas	Incorrectas mismo gesto	Incorrectas gesto equivocado	% Reconocimiento
GIRO DERECHA	50	48	2	0	96 %
GIRO IZQUIERDA	50	47	3	0	94 %
CUADRADO	50	45	5	0	90 %
ONDA	50	44	6	0	88 %

Tabla 10. Resultado de las pruebas (II).

A la vista de los resultados obtenidos el porcentaje de reconocimiento de los gestos es bastante elevado, teniendo el mayor porcentaje de acierto los gestos más sencillos y bajando dicho porcentaje los gestos un poco más complicados como son el cuadrado y la onda.

9 Conclusiones y trabajo futuro

9.1. Conclusiones

Después del trabajo realizado se pueden obtener las siguientes conclusiones:

- La condición física de la persona pueden llegar a influir en cuanto a que la señal parece presentar cierta componente de ruido cuando en realidad es el pulso de la mano. Si una persona acaba de realizar algún tipo de ejercicio la mano puede que tiemble más de lo habitual. También puede influir la edad ya que, una persona de más edad puede presentar un mayor temblor en su mano. Es necesario distinguir cuando en la señal hay un ruido debido a los componentes electrónicos de la IMU y cuando las perturbaciones son debidas a los temblores o vibraciones de la mano.
- La obtención de la posición únicamente por medio de los acelerómetros puede resultar bastante complicada debido al efecto de la gravedad. Con la ayuda de los giróscopos es posible reducir dicho efecto de forma bastante eficiente.
- En el caso de que se realizaran gestos con valores de pitch de $\pm 90^\circ$ la representación de la actitud por medio de los ángulos de Euler no sería fiable. Se deberían cambiar los cálculos de la actitud para representar la orientación con cuaterniones.
- Los errores debidos a los cálculos (integración de la aceleración, cálculo de la actitud,...) hacen imprescindible la corrección de la velocidad ya que el cálculo de la posición sería bastante impreciso. Dicha corrección funciona bien ya que el tiempo en que se utiliza no es demasiado grande. Si se quisiera obtener la posición y actitud de la mano durante un largo período de tiempo, habría que emplear alguna otra técnica de corrección como, por ejemplo, el filtro de Kalman.
- En movimientos más complejos, el uso de la lógica difusa puede quedar restringido ya que sería muy difícil emplear unas reglas de inferencia que se ajusten al reconocimiento de dichos gestos más complicados. Se podría utilizar en el caso de tener muchos más sensores y tener que decidir entre la información que de todos ellos se obtiene. En este

proyecto se ha desarrollado unas reglas muy sencillas que cubren el reconocimiento de gestos sencillos.

9.2. Trabajo futuro

- Realizar un estudio en profundidad de las vibraciones de la mano en personas con algún tipo de enfermedad motora (Parkinson,...).
- Realizar el proceso completo del reconocimiento de gestos de una forma on-line, de tal forma que la adquisición y el posterior reconocimiento se realicen de una forma inmediata.
- Ampliar el tiempo de los gestos para obtener la posición y actitud de la mano durante un largo período de tiempo. En este caso, la corrección de velocidad utilizada no funcionaría correctamente. Se implementaría un filtro de Kalman para corregir los errores.
- Fusionar el sistema de reconocimiento actual con uno basado en visión artificial. Se estudiaría la forma en que los dos se podrían complementar para obtener una mayor precisión en el reconocimiento.

10 Bibliografía

-
- [1] Mirabella, O.; Brischetto, M.; Mastroeni, G. "MEMS based gesture recognition", 2010 3rd Conference on Human System Interactions (HSI), Page(s): 599 – 604, Publication Year: 2010
 - [2] Ari Yosef Benbasat, "An Inertial Measurement Unit for User Interfaces" Máster thesis, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, Septiembre 2000
 - [3] Keogh, E.; Chu, S.; Hart, D.; Pazzani, M.; "An online algorithm for segmenting time series", Proceedings IEEE International Conference on Data Mining, 2001. ICDM 2001, Page(s): 289 – 296, Publication Year: 2001.
 - [4] N. Krahnstoever M. Yeasin R. Sharma, "Automatic Acquisition and Initialization of Kinematic Models", IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2001, Kauaii, Hawaii, USA.
 - [5] Jean-Christophe Lementec and Peter Bajcsy, "Recognition of Arm Gestures Using Multiple Orientation Sensors: Gesture Classification", 2004 IEEE Intelligent Transportation Systems Conference, Washington, D.C., USA, October 3-6, 2004.
 - [6] En Wei Huang, Li Chen Fu, "Gesture Stroke Recognition Using Computer Vision and Linear Accelerometer", 2008.
 - [7] XindongWu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, Dan Steinberg. "Top 10 algorithms in data mining", Springer-Verlag London Limited 2007,
 - [8] Michael Hoffman, Paul Varcholik, Joseph J. LaViola Jr. "Breaking the Status Quo: Improving 3D Gesture Recognition with Spatially Convenient Input Devices", IEEE Virtual Reality 2010 20 - 24 March, Waltham, Massachusetts, USA, 2010.
 - [9] Ruize Xu, Shengli Zhou, and Wen J. Li, "MEMS Accelerometer Based Non - Specific Specific -User Hand Gesture Recognition", IEEE 2011.Sensors Journal, IEEE Volume: PP, Issue:99.
 - [10] Shengli Zhou, Qing Shan, Fei Fei, Wen J. Li, Chung Ping Kwong, Patrick C. K. Wu, Bojun Meng, Christina K. H. Chan, Jay Y. J. Liou. "Gesture Recognition for Interactive Controllers Using MEMS Motion Sensors", Proceedings of the 2009

- 4th IEEE International Conference on Nano/Micro Engineered and Molecular Systems January 5-8, 2009, Shenzhen, China.
- [11] Alan H. F. Lam, Wen J. Li¹, Yunhui Liu, Ning Xi, "MIDS: Micro Input Devices System Using MEMS Sensors", Proceedings of the 2002 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems EPFL, Lausanne, Switzerland, October 2002.
- [12] Ahmad Akl, Shahrokh Valaee, "Accelerometer-based gesture recognition via Dynamic-Time Warping, affinity propagation & compressive sensing", IEEE 2010.
- [13] Jiayang Liu, Zhen Wang, Lin Zhong, Jehan Wickramasuriya, Venu Vasudevan, "uWave: Accelerometer-based Personalized Gesture Recognition and Its Applications", IEEE 2009.
- [14] Sy Bor Wang, Ariadna Quattoni, Louis-Philippe Morency, David Demirdjian, Trevor Darrell, "Hidden Conditional Random Fields for Gesture Recognition", Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06).
- [15] K. Hirota , H. A. Vu, P. Q. Le, C. Fatichah, Z. Liu, Y. Tang, M. L. Tangel, Z. Mu, B. Sun¹), F. Yan, D. Masano, O. Thet, M. Yamaguchi, F. Dong, and Y. Yamazaki, "Multimodal Gesture Recognition Based on Choquet Integral", 2011 IEEE International Conference on Fuzzy Systems June 27-30, 2011, Taipei, Taiwan.
- [16] Kosuke Makio, Yoshiki Tanaka, and Kuniaki Uehara, "Discovery of Skills from Motion Data", A. Sakurai et al. (Eds.): JSAI 2003/2004, LNAI 3609, pp. 266–282, 2007. ©Springer-Verlag Berlin Heidelberg 2007.
- [17] Pedro Neto, J. Norberto Pires, A. Paulo Moreira, "Accelerometer-Based Control of an Industrial Robotic Arm", The 18th IEEE International Symposium on Robot and Human Interactive Communication Toyama, Japan, Sept. 27-Oct. 2, 2009.
- [18] Xiubo Liang, Shun Zhang, Xiang Zhang, Weidong Geng, "Motion-based Perceptual User Interface, 2009 Third International Symposium on Intelligent Information Technology Application.

-
- [19] Tea Marasović, Vladan Papić, "Accelerometer-Based Gesture Classification Using Principal Component Analysis",
- [20] Narayanan C Krishnan, Prasanth Lade, Sethuraman Panchanathan, "Activity gesture spotting using a threshold model based on adaptive boosting", IEEE 2010
- [21] Carlos Delgado-Mata and Blanca Miriam Lee Cosio, "HMM and NN for Gesture Recognition", 2010 Electronics, Robotics and Automotive Mechanics Conference. IEEE 2010.
- [22] Xiaoyan Dang, Wei Wang, Kevin Wang, Mingzhi Dong, Liang Yin, "A user-independent Sensor Gesture interface for embedded device",
- [23] Jun-Ki Min, Bongwhan Choe, and Sung-Bae Cho, "A Selective Template Matching Algorithm for Short and Intuitive Gesture UI of Accelerometer-Built-in Mobile Phones", 2010 Second World Congress on Nature and Biologically Inspired Computing Dec. 15-17, 2010 in Kitakyushu, Fukuoka, Japan. IEEE 2010.
- [24] Jiahui Wu, Gang Pan, Daqing Zhang, Guande Qi, and Shijian Li, "Gesture Recognition with a 3-D Accelerometer", D. Zhang et al. (Eds.): UIC 2009, LNCS 5585, pp. 25–38, 2009. © Springer-Verlag Berlin Heidelberg 2009.
- [25] Thomas Schlömer, Benjamin Poppinga, Niels Henze, Susanne Boll, "Gesture Recognition with a Wii Controller",
- [26] Bastian Hartmann and Norbert Link, "Gesture Recognition with Inertial Sensors and Optimized DTW Prototypes", IEEE 2010.
- [27] Kazuya Murao, Tsutomu Terada, "A Motion Recognition Method by Constancy-Decision",
- [28] Doo Young Kwon and Markus Gross, "A Framework for 3D Spatial Gesture Design and Modeling Using a Wearable Input Device", IEEE 2007.
- [29] Liyanaarachchi Lekamalage Chamara Kasun, Wooi-Boon GOH, "ACCELEROMETER-BASED SWINGING GESTURE DETECTION FOR AN ELECTRONIC HANDBELL", 2011 IEEE 15th International Symposium on Consumer Electronics.

- [30] Mark Joselli, Esteban Clua, "gRmobile: A Framework for Touch and Accelerometer Gesture Recognition for Mobile Games", 2009 V III Brazilian Symposium on Digital Games and Entertainm, IEEE 2009.
- [31] Mohammad Helmi, S. M. T. AlModarresi, "Human Activity Recognition Using a Fuzzy Inference System", FUZZ-IEEE 2009, Korea, August 20-24, 2009. IEEE.
- [32] George M. Siouris. "Aerospace Avionics Systems. A modern synthesis". Academic Press Inc. ©1993.
- [33] Christopher J. Fisher. "Using an A ccelerometer for Inclination Sensing". Aplication Note AN-1057. Analog Devices. 2010.
- [34] Warren S. Flenniken IV, John H. Wall, David M. Bevly. "Characterization of Various IMU Error Sources and the Effect on Navigation Performance". Auburn University
- [35] Xsens. "MTi User Manual. MTi 10-series and MTi 100-series". Document MT0605P, Revision A, 26 Sept 2012.
- [36] Xsens. "MT Low-Level Communication Protocol Documentation". Document MT0101P, Revision O, 26 Sept 2012.
- [37] Mohinder S.Grewal, Lawrence R. Weill, Angus P. Andrews. "Global Positioning Systems. Inertial Navigation and I ntegration". Jhon Wiley and Sons, Inc. Publication. 2001.
- [38] H. Alemi Ardakani, T. J. Bridges. "Review of the 3-2-1 Euler Angles: a yaw–pitch–roll sequence". Department of Mathematics, University of Surrey, Guildford GU2 7XH UK. April 15, 2010.
- [39] Ji-Hwan Kim , Nguyen Duc Thang , Tae-Seong Kim, " 3D Hand Motion Tracking and Gesture Recognition Using a Data Glove" . IEEE International Symposium on Industrial Electronics (ISIE 2009) Seoul Olympic Parktel, Seoul , Korea July 5-8, 2009.
- [40] Jong Gwan Lim, Young Il Sohn, Dong Soo Kwon, "Real - Time Accelerometer Signal processing of end point detection and feature extraction for motion detection". Department of Mechanical Engineering, Korea Advanced Institute of Science and Technology.

-
- [41] Yilun Luo, Chi Chiu Tsang, Guanglie Zhang, Zhuxin Dong, Guangyi Shil, Sze Yin Kwok, Wen J. Lil, Philip H. W. Leong, and Ming YiuWong, "An Attitude Compensation Technique for a MEMS Motion Sensor Based Digital Writing Instrument. Proceedings of the 1st IEEE International Conference on Nano/Micro Engineered and Molecular Systems January 18 - 21, 2006, Zhuhai, China.
- [42] Edward Nelson Henderson, " An Inertial Measurement System for Hand and Finger Tracking". A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering Boise State University, December 2011.
- [43] Chul Woo Kang and Chan Gook Park, "Attitude Estimation with Accelerometers and Gyros Using Fuzzy Tuned Kalman Filter". Proceedings of European Control Conference 2009 - Budapest, Hungary, August 23 - 26, 2009.
- [44] Tullio Salmon Cinotti, Luigi Di Stefano, Giuseppe Raffa, Luca Roffia, Marina Pettinari, Martino Mola. "Dead reckoning supports stereo vision in pedestrians tracking". Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops 2006.
- [45] Hong Suh, Myung Kwan Park, Sanghoon Lee. "Design and Experiments of a 6-dimensional Spatial Tracker for Cheap Hand Motion Measurements". VECIMS 2003 - International Symposium on Virtual Environments, Human-Computer Interface and Measurement Systems Lugano, Switzerland. 21-29 July 2003.
- [46] Gonzalo Bailador, Sergio Guadarrama. "Robust gesture recognition using a Prediction-Error-Classification Approach".
- [47] Sangki Kim, Gunhyuk Park, Sunghoon Yim, Seungmoon Choi and Seungjin Choi. "Gesture-Recognizing Hand-Held Interface with Vibrotactile Feedback for 3D Interaction". IEEE Transactions on Consumer Electronics, Vol. 55, No. 3, August, 2009.
- [48] A.R. Jiménez, F. Seco, J.C. Prieto and J. Guevara. "Indoor Pedestrian Navigation using an INS/EKF framework for Yaw Drift Reduction and a Foot-

- mounted IMU". WPNC 2010: 7th Workshop on Positioning, Navigation and Communication 2010.
- [49] Hongliang Ren and Peter Kazanzides. "Investigation of Attitude Tracking Using an Integrated Inertial and Magnetic Navigation System for Hand-Held Surgical Instruments". IEEE/ASME Transactions on Mechatronics, VOL. 17, NO. 2, April 2012.
- [50] Antonio R. Jiménez, Fernando Seco, Francisco Zampella, José C. Prieto and Jorge Guevara. Article "PDR with a Foot-Mounted IMU and Ramp Detection". Sensors 2011, 11, pp. 9393-9410.
- [51] Demoz Gebre-Egziabher. "Design and Performance Analysis of a Low-cost Aided Dead Reckoning Navigator". Tesis Doctoral, Departamento de Aeronáutica, Universidad de Stanford, Febrero 2004.
- [52] Kim Jing Yang, Eun-Seok Choi, Wook Chang, Won-Chul Bang, Sung-Jung Cho, Jong Koo Oh, Joon Kee Cho, Dong Yoon. "A Novel Hand Gesture Input Device Based on Inertial Sensing Technique". The 30th Annual Conference of the IEEE Industrial Electronics Society, November 2 - 6, 2004, Busan, Korea. 2786-2791.
- [53] The Duy Bui, Long Thang Nguyen. "Recognizing Postures in Vietnamese Sign Language With MEMS Accelerometers". IEEE SENSORS JOURNAL, VOL. 7, NO. 5, 707-712. MAY 2007. .
- [54] John Kangchun Perng, Brian Fisher, Seth Hollar, Kristofer S. J. Pister. "Acceleration Sensing Glove (ASG)". Berkeley Sensor & Actuator Center University of California, Berkeley.
- [55] Belmar García García. "Diseño y construcción de un robot móvil controlado con técnicas de lógica difusa implementadas en una FPGA". Tesis de Máster. Escuela superior de Ingeniería mecánica y Eléctrica. Instituto Técnico Nacional. México DF. Septiembre 2012.
- [56] Andrés Jaramillo Botero. "Movimiento espacial de cuerpos rígidos".
- [57] Zadeh, L. A., Information and Control 1965, 8(3), 338-353.

11 Código Matlab

A continuación se detalla el código en Matlab empleado para la realización de las distintas funciones que componen el reconocimiento de gestos.

11.1. MTi_adquisicion_datos.m

```
function [imu]=MTi_adquisicion_datos

% En esta función se realizan la siguientes tareas:
%
%   - Configuración del MTi-300.
%   - Adquisición la de de lo datos calibrados de los acelerómetros
%     y gitóscopos.
%   - Almacenamiento en un archivo de los datos registrados.
%   - Visualización de los datos grabados.
%
% ENTRADA: -
%
% SALIDA:
%   - imu: estructura que contiene los datos de los
%     acelerómetros y giróscopos.
%-----
%-----

%% Introducir el número de serie del MTi-300

serial = 'NHZK-YKRX-5YX2-TGHZ-CZHK';

% El número de serie se puede tener almacenado en un fichero o
% introducirlo cuando se solicite.

if isempty(serial)
    serialFileName =
        fullfile(fileparts(which(mfilename)), 'serialkey.file');

    if exist(serialFileName, 'file')
        fid = fopen(serialFileName, 'r');
        serial = fread(fid, '*char')';
        fclose(fid);
    else
        fid = fopen(serialFileName, 'w');
        serial = input('Introduce el serial key:\n', 's');
        fprintf(fid, serial);
        fclose(fid);
    end
end

%% Activación del servidor activex

% Activa la API del SDK de Xsens en función de si windows
% es de 32 o 64 bits.
```

```
try
    switch computer
    case 'PCWIN'
        h = actxserver('xsensdeviceapi_com32.IXsensDeviceApi');
    case 'PCWIN64'
        h = actxserver('xsensdeviceapi_com64.IXsensDeviceApi');
    otherwise
        error('CMT:os','Unsupported OS');
    end
catch e
    fprintf('\n Please reinstall MT SDK or check manual,\n
    Xsens Device Api is not found.\n')
    rethrow(e);
end

fprintf( '\n ActiveXsens server - activated \n' );

% Registro del manejador de eventos. La lectura de los datos se
% realizará mediante el evento "onDataAvailable" que creará un
% evento siempre que haya un datos disponible para su lectura.

h.registerevent({'onDataAvailable',@eventhandlerXsens});

%% Establecimiento del número de serie.

serialOkay = h.XsControl_setSerialKey(serial);
if ~serialOkay
    fprintf('\n Serial key used: %s,\n check if this is the right
    one and try again.\n',serial);
    h.XsControl_close();
    return;
end

%% Escaneado de puertos del PC

% Se comprueba en qué puerto del PC se ha conectado el MTi-300
s = h.XsScanner_scanPorts(0,100,true);
if isempty(s)
    fprintf( '\n Connection ports - scanned - nothing found.\n' );
    return;
else
    fprintf( '\n Connection ports - scanned \n' );
    % Se abre el puerto
    deviceID = s{1,1};
    portS = s{1,3};
    baudRate = s{1,4};
    h.XsControl_openPort(portS,baudRate,0);
end

% Ahora se comprueba si hay algún otro dispositivo conectado
num_MTs = length(deviceID);
if num_MTs > 1
    fprintf('\n More than one device found, this script only uses
    the first one: %s\n',dec2hex(deviceID))
else
    fprintf('\n Device found: %s\n',dec2hex(deviceID))
end
```

```

device = h.XsControl_device(deviceID);

%% Configuración del dispositivo

% Se entra en el modo de configuración
h.XsDevice_gotoConfig(device);

% set output mode for this script:
outputConfig = h.XsDevice_outputConfiguration(device);

Fs = 400; % Frecuencia de muestreo.

% La primera columna indica el identificador de los datos
% y la segunda columna es la frecuencia. Si la frecuencia
% es 0 significa que el datos se enviará en todos los paquetes.

outputConfig_new = {'1020',0; % Contador de muestras
                    '4020',Fs; % Datos calibrados de aceleración
                    '8020',Fs}; % Datos calibrados de velocidad
                                angular

% Se carga la nueva configuración en el MTi.
outputConfig_new(:,1) = cellfun(@(x)
hex2dec(x),outputConfig_new(:,1),'UniformOutput',false);

h.XsDevice_setOutputConfiguration(device,outputConfig_new);

%% Creating log file
filename = [cd '\logfile_' dec2hex(deviceID) '.mtb'];
h.XsDevice_createLogFile(device,filename);

%% Reserva de espacio

% Se indica el número de muestras que se van a tomar.
maxSamples = 1200;

% Se reserva el espacio que ocuparan todos los datos
imu(1:maxSamples) =
    struct('acc',zeros(1,3),'gyr',zeros(1,3),'mag',zeros(1,3));

sampleCounter = NaN(maxSamples,1);
timeStamp = zeros(maxSamples,1);
iSample = 1; % contador de muestras.

%% Adquisición de datos

% Se cambia de estado y se entra en el estado de medida.
fprintf( '\n Activate measurement mode \n' );
h.XsDevice_gotoMeasurement(device);

% Comienza la grabación de los datos.
h.XsDevice_startRecording(device);
fprintf( '\n Logfile: %s created, start recording.\n',filename);

```

```
% read data
fprintf( '\n Reading data from devices... \n' );

% Se espera hasta que todos las muestras se hayan tomado
while (iSample <= maxSamples)

    pause(2)

end

fprintf( '\n ...done \n' );

%% Detención de la grabación de datos

% Se manda la orden de detener la grabación.
h.XsDevice_stopRecording(device);

% Se vuelve al estado de configuración.
h.XsDevice_gotoConfig(device);

% Se cierra el fichero log
h.XsDevice_closeLogFile(device)

%% Cierre de puerto

h.XsControl_closePort(portS);
h.XsControl_close();

%% Gráficas y grabación de los datos

dibujar_datos-inerciales(imu,Fs,0);
data=['..\archivo.mat'];
uisave('imu',data);

%% Función para los eventos

function eventhandlerXsens(varargin)
    % Solo cuando un nuevo paquete de datos está disponible.
    dataPacket = varargin{3}{1};

    if dataPacket && (iSample <= maxSamples)

        % Se comprueba si el contador de datos está dentro del
        % paquete de datos
        if h.XsDataPacket_containsPacketCounter(dataPacket)

            sampleCounter(iSample) =
                h.XsDataPacket_packetCounter(dataPacket);
```

```

end

% Se comprueba si hay datos calibrados de la velocidad
angular

if h.XsDataPacket_containsCalibratedGyroscopeData
    (dataPacket)

    calData =
        cell2mat(h.XsDataPacket_calibratedGyroscopeData
            (dataPacket));

    imu(iSample).gyr = calData(1:3);

end

% Se comprueba si hay datos calibrados de aceleración
if h.XsDataPacket_containsCalibratedAcceleration
    (dataPacket)

    calData2 =
        cell2mat(h.XsDataPacket_calibratedAcceleration
            (dataPacket));

    imu(iSample).acc = calData2(1:3);

end

end

iSample = iSample + 1;

end

end

```

11.2. dibujar_datos_inerciales

```

function dibujar_datos_inerciales(imu,frecuencia,all)

% En esta función crea 2 gráficas: una con los datos de
% aceleración de los tres acelerómetros y otra con los
% datos de velocidad angular de los tres giróscopos.
%
% ENTRADA:
% - imu: estructura que contiene los datos de acelerómetros
%       y giróscopos.
% - frecuencia: frecuencia a la que se muestrearon los
%       datos de imu.
% - all: Su valor indica como se presentan las gráficas.
%       * 0: Se agrupan en una sola figura.
%       * 1: Se crea una grafica con las aceleraciones y
%       otra con las velocidades angulares.
%

```



```

% SALIDA: -
%
% -----

ndatos=length(imu); % Longitud de los datos
tfinal=(ndatos-1)*(1/frecuencia); % Tiempo final de las muestras.
t=0:(1/frecuencia):tfinal; % eje de tiempo para las gráficas

% Se extraen los datos de la estructura. Esto se hace por
comodidad
% a la hora de manejar los datos de cada eje.
[accx,accy,accz,gyrx,gyry,gyrz] = obtener_datos(imu);

% Gráfica de las aceleraciones en los tres ejes.
if all
    figure;
    subplot(2,1,1);
else
    figure;
end

    plot(t,accx,'red',t,accy,'green',t,accz,'blue');
    hleg1 = legend('AccX','AccY','AccZ');
    set(hleg1,'Location','NorthEast');
    ylabel('m/s^2');
    xlabel('time(s)');
    title('Aceleración');

% Gráfica de la velocidad angular en los tres ejes.
if all
    subplot(2,1,2);
else
    figure;
end

%
plot(t,gyrx,'red',t,gyry,'green',t,gyrz,'blue');
hleg2 = legend('GyrX','GyrY','GyrZ');
set(hleg2,'Location','NorthEast');
ylabel('deg/s');
xlabel('time(s)');
title('Velocidad angular');

end

```

11.3. obtener_datos.m

```

function [accx,accy,accz,gyrx,gyry,gyrz] = obtener_datos(imu)

% Esta función obtiene los datos de los acelerómetros y
% los giróscopos que se encuentran almacenados en una

```

```

% variables (dato de entrada).
%
% ENTRADA:
%   - imu: estructura que contiene datos de los tres
%         acelerómetros y de los tres giróscopos.
%
% SALIDA:
%   - accx: datos del acelerómetro del eje x.
%   - accy: datos del acelerómetro del eje y.
%   - accz: datos del acelerómetro del eje z.
%   - gyrx: datos del giróscopo del eje x.
%   - gyry: datos del giróscopo del eje y.
%   - gyrz: datos del giróscopo del eje z.
%
%-----

ndatos=length(imu); % longitud de los datos

% Inicialización de las variables de salida

accx=zeros(1,ndatos);
accy=zeros(1,ndatos);
accz=zeros(1,ndatos);
gyrx=zeros(1,ndatos);
gyry=zeros(1,ndatos);
gyrz=zeros(1,ndatos);

% Se extraen los datos de imu y se almacenan en su
% variable correspondiente.

for k=1:ndatos

    accx(k)=imu(1,k).acc(1);
    accy(k)=imu(1,k).acc(2);
    accz(k)=imu(1,k).acc(3);

    % Los datos procedentes de los giróscopos se encuentran
    % en rad/s y se pasan a °/s.

    gyrx(k)=imu(1,k).gyr(1)*(360/(2*pi));
    gyry(k)=imu(1,k).gyr(2)*(360/(2*pi));
    gyrz(k)=imu(1,k).gyr(3)*(360/(2*pi));

end

end

```

11.4. filtro_media.m

```

function [datos_filtrados]=filtro_media( datos,frecuencia,dibujar )

% Se realiza un filtro de media a la señal de entrada y se
% devuelve la señal filtrada.

```

```

%
% ENTRADA:
%   - datos: contiene los datos de los acelerómetros o gióscopos.
%   - frecuencia: frecuencia a la que se obtuvieron los datos.
%   - dibujar: Representa en una gráfica los datos originales
%               y filtrados
%               * 0: No dibuja nada.
%               * 1: Dibuja los datos.
%
% SALIDA:
%   -datos_filtrados: contiene la señal filtrada.
% -----

ndatos=length(datos); % Longitud de los datos.
Ts=1/frecuencia;      % Periodo de muestreo
tfinal=(ndatos-1)*Ts; % Tiempo final de los datos
t=0:Ts:tfinal;        % Eje de tiempo para las graficas

n=12; %longitud del filtro;

muestras=zeros(1,n);
muestras(1,1:n)=mean(datos(1,1:n));
media=0;
datos_filtrados = zeros (1,ndatos);
j=12;

for k=1:ndatos

    muestras(1,j)=datos(1,k);
    media=mean(muestras);
    datos_filtrados(1,k)=media;

    if j==n
        j=1;
    else
        j=j+1;
    end
end

if dibujar==1
    figure;
    plot(t,datos,'red');
    hold on;
    plot(t,datos_filtrados,'blue');
    hleg1 = legend('Señal original','Señal filtrada');
    set(hleg1,'Location','NorthEast');
    ylabel('aceleración (m/s2)');
    xlabel('tiempo(s)');
    title('Filtro media');
end

end

```

11.5. segmentacion_1eje.m

```
function [dato_inicial,dato_final] = segmentacion_1eje( datos )

% En esta función se calcula el inicio y el final del gesto
% con las datos de un eje.
%
% ENTRADA:
% - datos: contiene las muestras de un acelerómetro
%
% SALIDA:
% - dato_inicial: número de la muestra en la que empieza.
%               el gesto.
% - dato_final: número de la muestra en la que acaba el gesto.
% -----

ndatos=length(datos); % Longitud de los datos

% Inicialización de variables.
dato_inicial=0;
dato_final=0;

% Definición de los umbrales inicial y final
umbral_inicio=0.7;
umbral_final=1;

n=40; % Espacio entre las muestras.
k=1;  % Número de muestra por la que se empieza a recorrer los
      % datos.

%% Cálculo del inicio del gesto

% Se recorren todas las muestras desde la primera y se calcula
% el valor absoluto de la diferencia entre una muestra k y
% la muestra k+n. Si el resultado es mayor al umbral de inicio
% entonces el inicio del gesto será la muestra k

while k<(ndatos-n)

    incremento=abs(datos(1,k+n))-abs(datos(1,k));

    if incremento>umbral_inicio
        dato_inicial=k;
        break;
    end

    k=k+1;
end

%% Cálculo del final del gesto

% Se recorren todas las muestras desde la última y se calcula
% el valor absoluto de la diferencia entre una muestra k y
% la muestra k-n. Si el resultado es mayor al umbral de final
```

```
% entonces el final del gesto será la muestra k

k=ndatos;
while k>n

    incremento=abs(datos(1,k-n))-abs(datos(1,k));

    if incremento>umbral_final
        dato_final=k;
        break;
    end
    k=k-1;

end

%% Comprobación de error

% En caso de que el inicio del gesto sea mayor al final, se
% se definirán el inicio y el final como datos no calculables.

if dato_inicial >= dato_final

    dato_inicial=NaN;
    dato_final=NaN;

end

end
```

11.6. segmentacion_3ejes.m

```
function [ inicio,fin ] = segmentacion_3ejes( accx,accy,accz)

% Esta función calcula el inifio y el final del gesto en las
% tres señales de los acelerómetros.
%
% ENTRADA:
% - accx: aceleración en el ejex.
% - accy: aceleración en el ejey.
% - accz: aceleración en el ejez.
%
% SALIDA:
% - inicio: número de la muestra en la que empieza el gesto.
% - final: número de la muestra en la que acaba el gesto.
% -----

% Se calculan el inicio y el final cada eje por separado

[ iniciox,finx ] = segmentacion_1eje (accx);
[ inicioy,finy ] = segmentacion_1eje (accy);
[ inicioz,finz ] = segmentacion_1eje (accz);
```

```

% Se calcula el inicio y el final del gesto en los tres ejes.
% El inicio será el mínimo valor de los tres inicios y el
% final será el máximo valor de los tres finales.

inicio=min([iniciox inicioy inicioz]);
fin=max([finx finy finz]);

end

```

11.7. posicionamiento.m

```

function
[maxx,minx,maxy,miny,maxz,minz,maxpitch,minpitch,maxroll,minroll,posx_
final,posy_final,posz_final]=posicionamiento( imu,frecuencia,dibujar )

% Esta función calcula la posición de la mano en base a los
% datos de aceleración y velocidad angular.
%
% ENTRADA
% - imu: estructura que contiene los datos de aceleración
%       y velocidad angular.
% - frecuencia: frecuencia a la que se muestrearon los datos.
% - dibujar: Representa en una gráfica los datos originales
%            y filtrados
%           * 0: No dibuja nada.
%           * 1: Dibuja los datos.
%
% SALIDA
% - maxx: valor máximo en centímetros en el eje x.
% - minx: valor mínimo en centímetros en el eje x.
% - maxy: valor máximo en centímetros en el eje y.
% - miny: valor mínimo en centímetros en el eje y.
% - maxz: valor máximo en centímetros en el eje z.
% - minz: valor mínimo en centímetros en el eje z.
% - maxpitch: valor máximo en grados en pitch.
% - minpitch: valor mínimo en grados en pitch.
% - maxroll: valor máximo en grados en roll.
% - minroll: valor mínimo en grados en roll.
% - posx_final: última posición en centímetros en el eje x.
% - posy_final: última posición en centímetros en el eje x.
% - posz_final: última posición en centímetros en el eje x.
%
% -----

%% Inicialización de variables

acc_nav=zeros(3,ndatos);
acc_body=zeros(3,ndatos);
acc_body2=zeros(3,ndatos);

roll = zeros(1,ndatos);
pitch = zeros(1,ndatos);
yaw = zeros(1,ndatos);

w_nav = zeros(3,ndatos);

```

```
roll_nav = zeros(1,ndatos);
pitch_nav = zeros(1,ndatos);
yaw_nav = zeros(1,ndatos);

vel_accx_nav = zeros(1,ndatos);
vel_accy_nav = zeros(1,ndatos);
vel_accz_nav = zeros(1,ndatos);

pos_accx_nav = zeros(1,ndatos);
pos_accy_nav = zeros(1,ndatos);
pos_accz_nav = zeros(1,ndatos);

correccion_velx_nav = zeros(1,ndatos);
correccion_vely_nav = zeros(1,ndatos);
correccion_velz_nav = zeros(1,ndatos);

vel_accx_nav_corregida = zeros(1,ndatos);
vel_accy_nav_corregida = zeros(1,ndatos);
vel_accz_nav_corregida = zeros(1,ndatos);

pos_accx_nav_corregida = zeros(1,ndatos);
pos_accy_nav_corregida = zeros(1,ndatos);
pos_accz_nav_corregida = zeros(1,ndatos);

posx_final= 0;
posy_final= 0;
posz_final= 0;

maxx=0;
maxy=0;
maxz=0;

minx=0;
miny=0;
minz=0;

G=[0;0;-9.80665];

%% Obtención de los datos

% Se obtienen los datos de los acelerómetros y los
% giróscopos por separado.
[accx,accy,accz,gyrx,gyry,gyrz]=obtener_datos(imu);

%% Coordenadas de body

% Se convierten los ejes a las coordenadas de body
accy=-accy;
accz=-accz;
gyry=-gyry;
gyrz=-gyrz;

%% Filtrado
```

```

% Se realiza un filtro de media para eliminar el ruido.
[accx_bf]=filtro_medio( accx,frecuencia,0 );
[accy_bf]=filtro_medio( accy,frecuencia,0 );
[accz_bf]=filtro_medio( accz,frecuencia,0 );
[gyrx_bf]=filtro_medio( gyrx,frecuencia,0 );
[gyry_bf]=filtro_medio( gyry,frecuencia,0 );
[gyrz_bf]=filtro_medio( gyrz,frecuencia,0 );

%% Segmentación

% Segmentación de los datos. Se obtiene el inicio y el fin
% del gesto.

[inicio,fin] = segmentacion_3ejes( accx_bf,accy_bf,accz_bf);
ndatos=fin-inicio+1;

% Se recortan las señales de los acelerómetros y giroscopos

accx_b(1,:)=accx_bf(1,inicio:fin);
accy_b(1,:)=accy_bf(1,inicio:fin);
accz_b(1,:)=accz_bf(1,inicio:fin);

gyrx_b(1,:)=gyrx_bf(1,inicio:fin);
gyry_b(1,:)=gyry_bf(1,inicio:fin);
gyrz_b(1,:)=gyrz_bf(1,inicio:fin);

%% Ángulos iniciales

% Se calcula el valor inicial de los ángulos de Euler
pitch_inicial=atand(accx_inicial/
                    sqrt(accy_inicial^2+accz_inicial^2));
roll_inicial=-atand(accy_inicial/
                    sqrt(accx_inicial^2+accz_inicial^2));
yaw_inicial=0;

roll(1,1)=roll_inicial;
pitch(1,1)=pitch_inicial;
yaw(1,1)=yaw_inicial;

roll_nav(1,1)=roll_inicial;
pitch_nav(1,1)=pitch_inicial;
yaw_nav(1,1)=yaw_inicial;

maxpitch=pitch_nav(1,1);
minpitch=pitch_nav(1,1);
maxroll=roll_nav(1,1);
minroll=roll_nav(1,1);

%% Cambio de coordenadas
for k=2:ndatos

    % Obetnción de los ángulos en body
    pitch(1,k)= pitch(1,k-1) + (gyry_b(1,k)*Ts);
    roll(1,k) = roll(1,k-1) + (gyrx_b(1,k)*Ts);

```



```
yaw(1,k) = yaw(1,k-1) + (gyrz_b(1,k)*Ts);

% Matriz para pasar a ángulos de navegación
A11=1;
A12=sind(roll_nav(1,k-1))*tand(pitch_nav(1,k-1));
A13=cosd(roll_nav(1,k-1))*tand(pitch_nav(1,k-1));
A21=0;
A22=cosd(roll_nav(1,k-1));
A23=-sind(roll_nav(1,k-1));
A31=0;
A32=sind(roll_nav(1,k-1))/cosd(pitch_nav(1,k-1));
A33=cosd(roll_nav(1,k-1))/cosd(pitch_nav(1,k-1));

A=[A11 A12 A13;
    A21 A22 A23;
    A31 A32 A33];

% Velocidades angulares en navigation
w_nav(:,k)=A*[gyrx_b(1,k);gyry_b(1,k);gyrz_b(1,k)];

% Ángulos en navigation
roll_nav(1,k) = roll_nav(1,k-1)+ w_nav(1,k)*Ts;
pitch_nav(1,k)= pitch_nav(1,k-1)+ w_nav(2,k)*Ts;
yaw_nav(1,k) = yaw_nav(1,k-1)+ w_nav(3,k)*Ts;

% Se construye la matriz de cambio de coordenadas DCM
C11=cosd(yaw_nav(1,k))*cosd(pitch_nav(1,k));

C12=-(sind(yaw_nav(1,k))*cosd(roll_nav(1,k)))+
    (cosd(yaw_nav(1,k))*sind(pitch_nav(1,k))*
    sind(roll_nav(1,k)));

C13=(sind(yaw_nav(1,k))*sind(roll_nav(1,k)))+
    (cosd(yaw_nav(1,k))*sind(pitch_nav(1,k))
    *cosd(roll_nav(1,k)));

C21=sind(yaw_nav(1,k))*cosd(pitch_nav(1,k));

C22=(cosd(yaw_nav(1,k))*cosd(roll_nav(1,k)))+
    (sind(yaw_nav(1,k))*sind(pitch_nav(1,k))*
    sind(roll_nav(1,k)));

C23=-(cosd(yaw_nav(1,k))*sind(roll_nav(1,k)))+
    (sind(yaw_nav(1,k))*sind(pitch_nav(1,k))*
    cosd(roll_nav(1,k)));

C31=-sind(pitch_nav(1,k));

C32=cosd(pitch_nav(1,k))*sind(roll_nav(1,k));

C33=cosd(pitch_nav(1,k))*cosd(roll_nav(1,k));

DCM=[C11 C12 C13;
    C21 C22 C23;
    C31 C32 C33];

% Matriz con todas las aceleraciones
acc_body(:,k)=[accx_b(1,k);accy_b(1,k);accz_b(1,k)];
```

```

% Se calcula el efecto de la gravedad en los acelerómetros
g_body(:,k)=DCM'*G;

% Se realiza la corrección de la gravedad sobre las
  aceleraciones
acc_cor(:,k)= acc_body(:,k) - g_body(:,k);

% Se pasa a aceleraciones de navegación
acc_nav(:,k)= DCM * acc_cor(:,k);

% Se calcula la velocidad de navegación derivando la
  aceleración
vel_accx_nav(1,k)=vel_accx_nav(1,k-1) + (acc_nav(1,k) * Ts);
vel_accy_nav(1,k)=vel_accy_nav(1,k-1) + (acc_nav(2,k) * Ts);
vel_accz_nav(1,k)=vel_accz_nav(1,k-1) + (acc_nav(3,k) * Ts);

% Se calcula la posicion de navegacion
pos_accx_nav(1,k)= pos_accx_nav(1,k-1) + vel_accx_nav(1,k)*Ts;
pos_accy_nav(1,k)= pos_accy_nav(1,k-1) + vel_accy_nav(1,k)*Ts;
pos_accz_nav(1,k)= pos_accz_nav(1,k-1) + vel_accz_nav(1,k)*Ts;

% se calculan los valores máximo y mínimos de pitch y roll
if pitch_nav(1,k)> maxpitch
    maxpitch=pitch_nav(1,k);
elseif pitch_nav(1,k)< minpitch
    minpitch=pitch_nav(1,k);
end

    if roll_nav(1,k)> maxroll
        maxroll=roll_nav(1,k);
    elseif roll_nav(1,k)< minroll
        minroll=roll_nav(1,k);
    end

end

%% Corrección velocidad

for k=2:ndatos

    % Se calcula el valor de la correccion
    correccion_velx_nav(1,k) =
        (k*Ts)*(vel_accx_nav(1,ndatos)/tfinal);

    correccion_vely_nav(1,k) =
        (k*Ts)*(vel_accy_nav(1,ndatos)/tfinal);

    correccion_velz_nav(1,k) =
        (k*Ts)*(vel_accz_nav(1,ndatos)/tfinal);

    % Se corrige el error en velocidad
    vel_accx_nav_corregida (1,k) =
        vel_accx_nav(1,k)-correccion_velx_nav(1,k);

```

```
vel_accy_nav_corregida (1,k) =  
    vel_accy_nav(1,k)-correccion_vely_nav(1,k);  
  
vel_accz_nav_corregida (1,k) =  
    vel_accz_nav(1,k)-correccion_velz_nav(1,k);  
  
% Se vuelve a calcular la posición  
pos_accx_nav_corregida(1,k)= pos_accx_nav_corregida(1,k-1) +  
    vel_accx_nav_corregida(1,k)*Ts;  
  
pos_accy_nav_corregida(1,k)= pos_accy_nav_corregida(1,k-1) +  
    vel_accy_nav_corregida(1,k)*Ts;  
  
pos_accz_nav_corregida(1,k)= pos_accz_nav_corregida(1,k-1) +  
    vel_accz_nav_corregida(1,k)*Ts;  
  
% Se registran las máximas y mínimas posiciones en los tres  
ejes  
if pos_accx_nav_corregida(1,k)> maxx  
    maxx=pos_accx_nav_corregida(1,k);  
elseif pos_accx_nav_corregida(1,k)< minx  
    minx=pos_accx_nav_corregida(1,k);  
end  
  
if pos_accy_nav_corregida(1,k)> maxy  
    maxy=pos_accy_nav_corregida(1,k);  
elseif pos_accy_nav_corregida(1,k)< miny  
    miny=pos_accy_nav_corregida(1,k);  
end  
  
if pos_accz_nav_corregida(1,k)> maxz  
    maxz=pos_accz_nav_corregida(1,k);  
elseif pos_accz_nav_corregida(1,k)< minz  
    minz=pos_accz_nav_corregida(1,k);  
end  
end  
  
% Los datos se convierten a centímetros  
posx_final= pos_accx_nav_corregida(1,ndatos)*100;  
posy_final= pos_accy_nav_corregida(1,ndatos)*100;  
posz_final= pos_accz_nav_corregida(1,ndatos)*100;  
  
maxx = maxx*100;  
minx = minx*100;  
maxy = maxy*100;  
miny = miny*100;  
maxz = maxz*100;  
minz = minz*100;  
  
% Se registra el máximo y el mínimo valor comparándolo con los  
valores iniciales  
  
maxpitch=maxpitch-pitch_nav(1,1);  
minpitch=minpitch-pitch_nav(1,1);  
maxroll=maxroll-roll_nav(1,1);  
minroll=minroll-roll_nav(1,1);
```

```

%% Se dibujan los datos en diversas gráficas

if dibujar==1

    data=['Eje x (cm):    max = ',num2str(maxx),' min = ',num2str(minx)]; disp(data);

    data=['Eje y (cm):    max = ',num2str(maxy),' min = ',num2str(miny)]; disp(data);

    data=['Eje z (cm):    max = ',num2str(maxz),' min = ',num2str(minz)]; disp(data);

    data=['Giro (grados) en pitch:    max = ',num2str(maxpitch),' min = ',num2str(minpitch)]; disp(data);

    data=['Giro (grados) en roll:    max = ',num2str(maxroll),' min = ',num2str(minroll)]; disp(data);

    Ts=1/frecuencia;
    tfinal2=(ndatos-1)*Ts;
    t2=0:Ts:tfinal2;

    % angulos body
    figure;
    plot(t2,pitch,'red');
    hold on;
    plot(t2,roll,'green');
    hold on;
    plot(t2,yaw,'blue');
    ylabel('grados');
    xlabel('time(s)');
    hleg2 = legend('Pitch','Roll','Yaw');
    set(hleg2,'Location','NorthEast');
    title('Angulos en los tres ejes');

    % angulos navegacion
    figure;
    plot(t2,pitch_nav,'red');
    hold on;
    plot(t2,roll_nav,'green');
    hold on;
    plot(t2,yaw_nav,'blue');
    ylabel('grados');
    xlabel('time(s)');
    hleg2 = legend('Pitch','Roll','Yaw');
    set(hleg2,'Location','NorthEast');
    title('Angulos en los tres ejes de navegacion');

    % Aceleraciones corregidas
    figure;
    subplot(3,1,1);
    plot(t2,acc_cor(1,:), 'red');
    hold on;
    plot(t2,accx_b, 'blue');
    hold on;

```

```

plot(t2,g_body(1,:), 'green');
ylabel('m/s2');
xlabel('time(s)');
hleg2 = legend('AccX corregida', 'AccX', 'gravedad');
set(hleg2, 'Location', 'NorthEast');
title('Aceleración Eje X body');

subplot(3,1,2);
plot(t2,acc_cor(2,:), 'red');
hold on;
plot(t2,accy_b, 'blue');
hold on;
plot(t2,g_body(2,:), 'green');
ylabel('m/s2');
xlabel('time(s)');
hleg2 = legend('AccY corregida', 'AccY', 'gravedad');
set(hleg2, 'Location', 'NorthEast');
title('Aceleración Eje Y body');

subplot(3,1,3);
plot(t2,acc_cor(3,:), 'red');
hold on;
plot(t2,accz_b, 'blue');
hold on;
plot(t2,g_body(3,:), 'green');
ylabel('m/s2');
xlabel('time(s)');
hleg2 = legend('AccZ corregida', 'AccZ', 'gravedad');
set(hleg2, 'Location', 'NorthEast');
title('Aceleración Eje Z body');

% Aceleraciones de navegacion
figure;
subplot(3,3,1);
plot(t2,acc_nav(1,:), 'red');
hold on;
plot(t2,acc_cor(1,:), 'blue');
% hold on;
% plot(t2,acc_body2(1,:), 'green');
ylabel('m/s2');
xlabel('time(s)');
title('Aceleración Eje X navegación');

subplot(3,3,2);
plot(t2,acc_nav(2,:), 'red');
hold on;
plot(t2,acc_cor(2,:), 'blue');
% hold on;
% plot(t2,acc_body2(2,:), 'green');
ylabel('m/s2');
xlabel('time(s)');
title('Aceleración Eje Y navegación');

subplot(3,3,3);
plot(t2,acc_nav(3,:), 'red');
hold on;
plot(t2,acc_cor(3,:), 'blue');
% hold on;
% plot(t2,acc_body2(3,:), 'green');
ylabel('m/s2');

```

```

xlabel('time(s)');
title('Aceleración Eje Z navegación');

% Velocidades
subplot(3,3,4);
plot(t2,vel_accx_nav,'blue');
hold on;
plot(t2,correccion_velx_nav,'green');
hold on;
plot(t2,vel_accx_nav_corregida,'red');
ylabel('m/s');
xlabel('time(s)');
title('Velocidad corregida Eje X');

subplot(3,3,5);
plot(t2,vel_accy_nav,'blue');
hold on;
plot(t2,correccion_vely_nav,'green');
hold on;
plot(t2,vel_accy_nav_corregida,'red');
ylabel('m/s');
xlabel('time(s)');
title('Velocidad corregida Eje Y');

subplot(3,3,6);
plot(t2,vel_accz_nav,'blue');
hold on;
plot(t2,correccion_velz_nav,'green');
hold on;
plot(t2,vel_accz_nav_corregida,'red');
ylabel('m/s');
xlabel('time(s)');
title('Velocidad corregida Eje Z');

% Posiciones
subplot(3,3,7);
plot(t2,pos_accx_nav,'blue');
hold on;
plot(t2,pos_accx_nav_corregida,'red');
ylabel('m');
xlabel('time(s)');
title('Posicion corregida Eje X');

subplot(3,3,8);
plot(t2,pos_accy_nav,'blue');
hold on;
plot(t2,pos_accy_nav_corregida,'red');
ylabel('m');
xlabel('time(s)');
title('Posicion corregida Eje Y');

subplot(3,3,9);
plot(t2,pos_accz_nav,'blue');
hold on;
plot(t2,pos_accz_nav_corregida,'red');
ylabel('m');
xlabel('time(s)');
title('Posicion corregida Eje Z');

```

```
end  
end
```